Report on

RideHive - Bike and Scooter Rental System

- By

Samruddhi Chavan | N01604191
Saloni Patel | N01603895
Sruthi Pandiath | N01618202
Mitali Sisodia | N01621572

Table of Contents

1. Introduction 3
1.1. Project Overview 3
1.2. Objectives 3
2. Problem Statement 3
3. Proposed Solution 3
4. Technologies Used 4
5. Database Schema 4
6. Features 5
7. UML Diagrams 6
7.1. Class Diagram 6
7.2. Sequence Diagram 8
8. Deployment Process Using Podman 10
9. Screenshots 11
10. Conclusion 31

1. Introduction

RideHive is a web-based bike and scooter rental system designed to provide affordable and flexible transportation for urban users. It streamlines vehicle rentals for users, vendors, and admins with features like secure payments, efficient fleet management, and modular scalability.

1.1. Project Overview

RideHive is a web-based platform developed to provide convenient access to short-term bike and scooter rentals. Catering to users, vendors, and administrators, the system leverages Laravel and other technologies to ensure functionality, security, and scalability. By addressing urban transportation needs, RideHive offers a streamlined solution for affordable and flexible vehicle rentals.

1.2. Objectives

The primary objectives of the RideHive platform are as follows:

- Simplify the process of renting bikes and scooters for users.
- Empower vendors to efficiently manage vehicle listings and bookings.
- Facilitate administrators in monitoring and regulating platform operations.
- Ensure secure and user-friendly online payments through **Stripe integration**.
- Develop a modular and scalable architecture to handle future enhancements.

2. Problem Statement

With the increasing demand for temporary and cost-effective transportation solutions, RideHive fills a critical gap in the market. Urban dwellers, students, tourists, and professionals benefit from a flexible rental service that does not require vehicle ownership.

3. Proposed Solution

- **Urban Need:** In cities, many individuals require transportation for short durations. Renting is often more practical and economical than ownership.
- **Vendor Opportunity:** Vendors can list underutilized vehicles, generating passive income.
- **Streamlined Operations:** By centralizing transactions, bookings, and listings, RideHive ensures ease of use for all stakeholders.
- **Technology-Driven Efficiency:** The platform's Laravel-based backend and Stripe integration provide a secure, responsive, and scalable solution.

4. Technologies Used

- PHP: Core programming language for backend logic.
- MySQL: Database management system.
- HTML/CSS(Tailwind)/JavaScript: Frontend technologies for user interface.
- Autoloading**:** Composer facilitates class autoloading, minimizing boilerplate code and ensuring efficient application setup.
- Database Migration**:** Laravel's migration system manages database version control, making schema changes collaborative and conflict-free
- Laravel is a PHP-based framework that adheres to the Model-View-Controller (MVC) architecture, ensuring clean code and modular development.

5. Database Schema

- **Users Table**
  - Fields: user_id, name, email, password, role, status, created_at
  - Relationships: One-to-many with bookings and feedback.

- **Vehicles Table**
  - Fields: vehicle_id, vendor_id, type, model, price_per_day, status, created_at
  - Relationships: Belongs to vendors; one-to-many with bookings and availability.

- **Bookings Table**
  - Fields: booking_id, user_id, vehicle_id, start_date, end_date, total_cost, status
  - Relationships: Belongs to users and vehicles.

- **Payments Table**
  - Fields: payment_id, booking_id, amount, status, payment_date
  - Relationships: Belongs to bookings; ensures secure transactions.

- **Availability Table**
  - Fields: availability_id, vehicle_id, start_date, end_date, status
  - Relationships: Tracks vehicle availability, linking to bookings when applicable.

- **Relationships Summary**
  - Users → Bookings → Vehicles → Payments: Tracks user interaction from browsing to payment.
  - Vendors → Vehicles → Availability: Links vendors with their fleet and rental status.

6. Features

RideHive consists of distinct modules designed for different user roles:

Admin Module

- Manage users, vendors, vehicles, and bookings.
- Monitor transactions via dashboards.

User Module

- Search and filter available vehicles by type, price, and availability.
- Book and pay for rentals using Stripe for secure online payments.

Vendor Module

- List, update, and remove vehicles with pricing and availability details.
- Track bookings and customer interactions.

Payment Module

- Integrated with Stripe API, enabling secure and automated payment processing.
- Handles payment status (successful, failed, pending) and logs transaction details.

7. UML Diagrams

Unified Modeling Language (UML) is a standardized modeling language used in software engineering. It helps visualize, specify, and document the architecture and design of software systems. UML includes several diagram types such as class diagrams, sequence diagrams, use-case diagrams, and more.

7.1. Class Diagram

1. **Purpose**:
    - Represents the structure of the system.
    - Shows classes, their attributes, methods, and relationships.
2. **Elements**:
    - **Classes**: Represent entities with attributes (data) and methods (functions).
    - **Relationships**:
        - **Association**: Represents "uses" or "has-a" relationships (e.g., Users "owns" Vehicles).
        - **Multiplicity**: Specifies how many instances participate in the relationship (e.g., 0..* means "zero or more").
        - **Inheritance**: Shows generalization-specialization relationships (not explicitly shown in this diagram).
        - **Aggregation/Composition**: Denotes "part-of" relationships (not shown here).
3. **Analysis of the Class Diagram**:
    - The system manages users, vehicles, bookings, availability, payments, and feedback.
    - Key Entities:
        - **Users**: Handles user details and operations like login, registration, and booking.
        - **Vehicles**: Represents rentable vehicles with attributes like type, model, price, and status.
        - **Bookings**: Links users to vehicles through reservation details.
        - **Availability**: Tracks when vehicles are available.
        - **Payments**: Records payment details for bookings.
        - **Feedback**: Stores user reviews for bookings.
    - Relationships:
        - Users can own Vehicles.
        - Vehicles have Availability and are linked to Bookings.
        - Bookings are associated with Payments and Feedback.

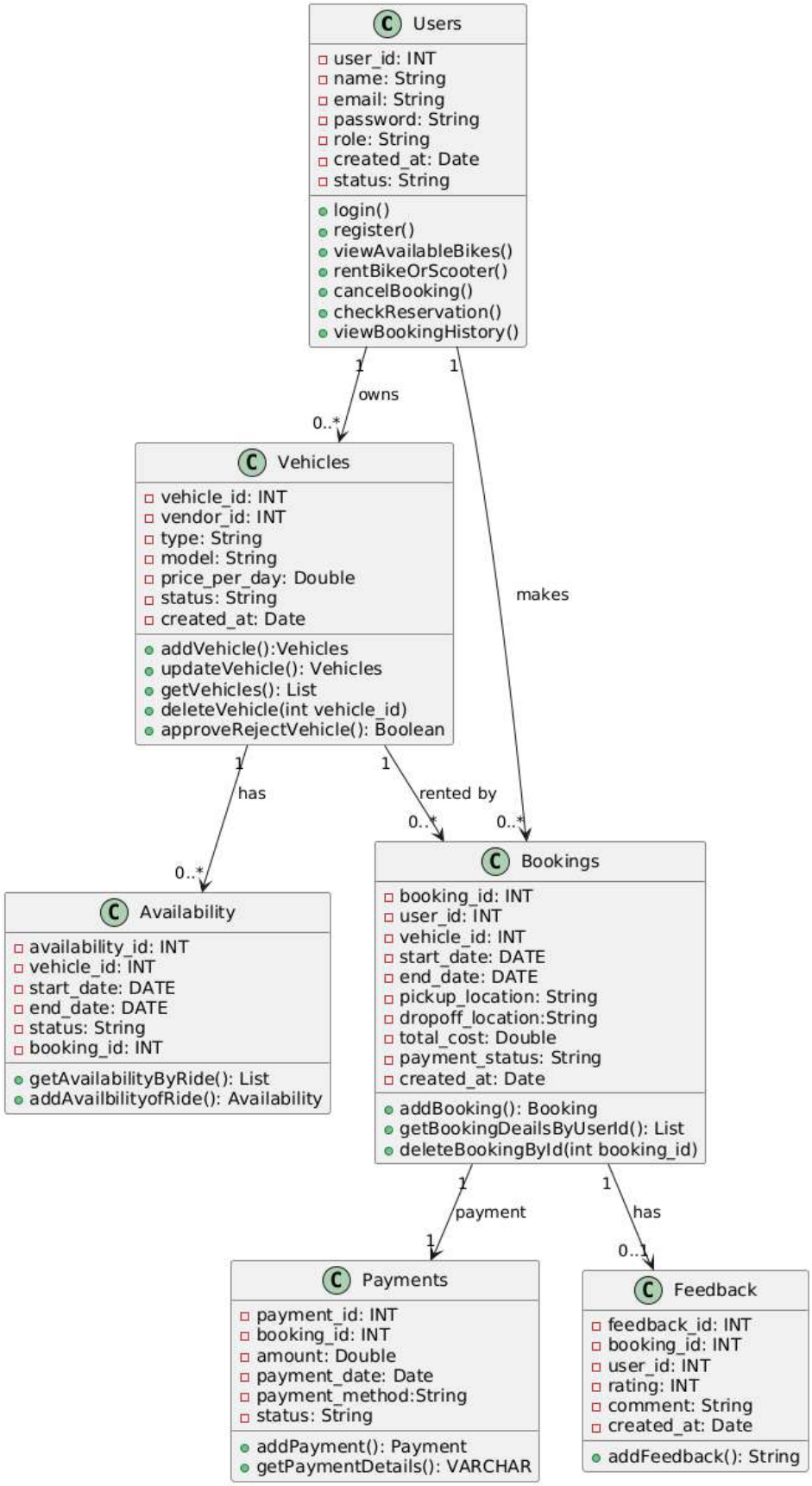


Fig.1. Class Diagram

7.2. Sequence Diagram

1. **Purpose**:
    - Illustrates how processes or operations are carried out.
    - Displays interactions between system components in a time-sequential manner.
2. **Elements**:
    - **Actors**: Represent users or systems interacting with the software (e.g., Vendor, User, Admin).
    - **Lifelines**: Show the lifespan of an object during interactions.
    - **Messages**: Depict communication between objects (synchronous or asynchronous).
    - **Execution Occurrence**: Indicates when an object is performing an operation.
3. **Analysis of the Sequence Diagram (Image 2)**:
    - Key Use Cases:
        1. **User and Vendor Registration and Login**:
            - Users and vendors register and validate credentials via authorization emails.
            - Login is confirmed once credentials are validated.
        2. **Vehicle Management**:
            - Vendors can add, update, and delete vehicle records, which are reviewed for approval.
        3. **Vehicle Booking**:
            - Users search for available vehicles, request bookings, and make payments.
            - Payment status is updated, and booking confirmation is sent.
        4. **Vehicle Availability and Cancellation**:
            - Availability is queried when a user checks for a ride.
            - Bookings can be canceled, updating the vehicle's availability.
        5. **Admin Management**:
            - Admin retrieves payment transactions and user/vendor details for oversight.

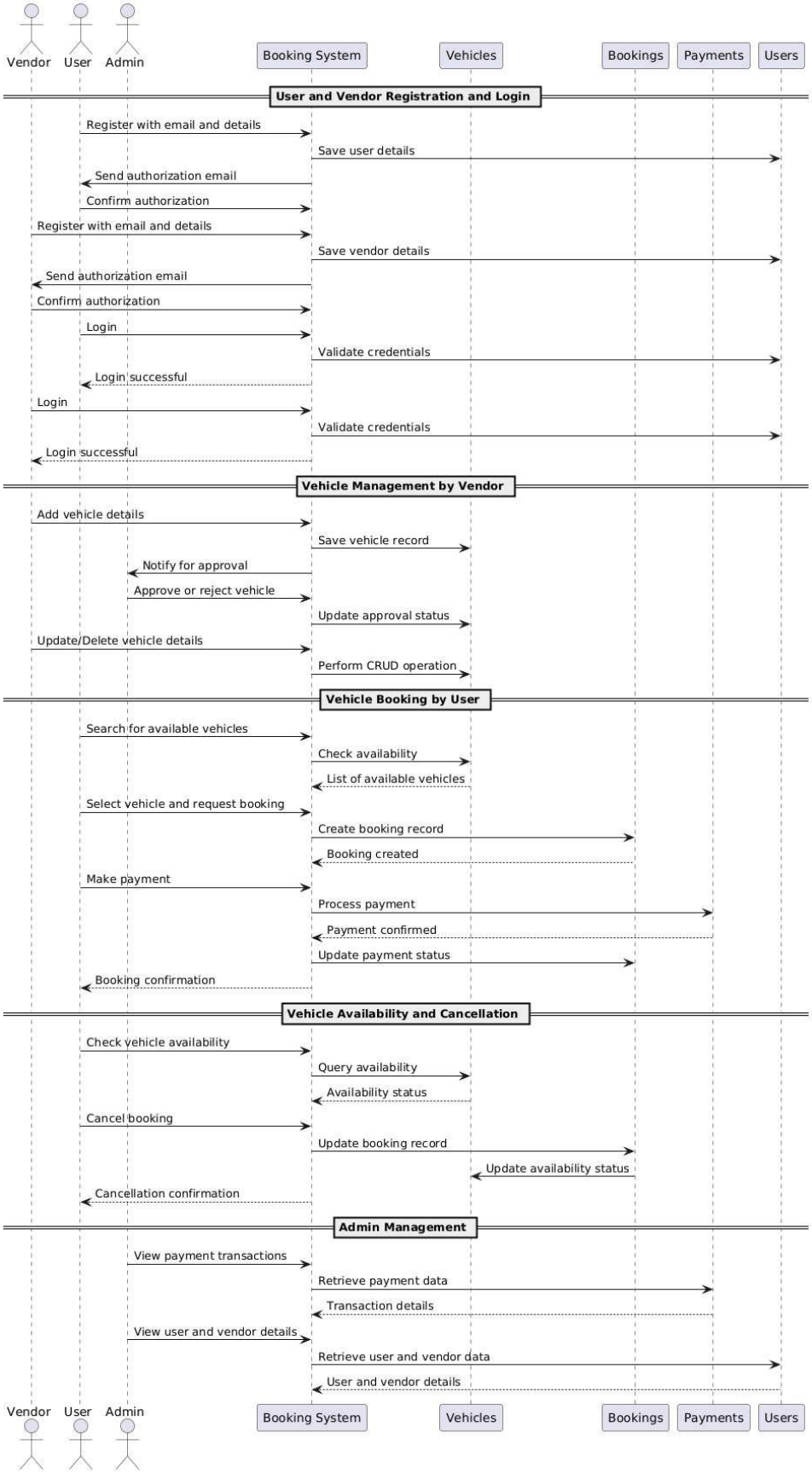


Fig.2. Sequence Diagram

8. Deployment Process Using Podman

For deploying our Federal Parliament Legislation Process Management System, we used **Podman** to manage containerized applications due to its enhanced security and compatibility with Docker commands. Podman enables rootless containers, allowing applications to run without root privileges, which improves security.

Dockerfile: Defines the application environment, including the PHP version and necessary dependencies.

FROM php:8.1-apache – This line creates an image of apache 8.1 to run the php application

RUN docker-php-ext-install pdo pdo_mysql – This line asks the container to install mysql-pdo

COPY src/ /var/www/html/ - This line copies all the content from src/ to the root directory of container

COPY .env /var/www/html/.env - This line copies all the content from .env file to the root directory of container

WORKDIR /var/www/html – This line declares this as its working directory

EXPOSE 80 – This line specify that container will listen on port **80**

docker-compose.yml:

This file defines two main services:

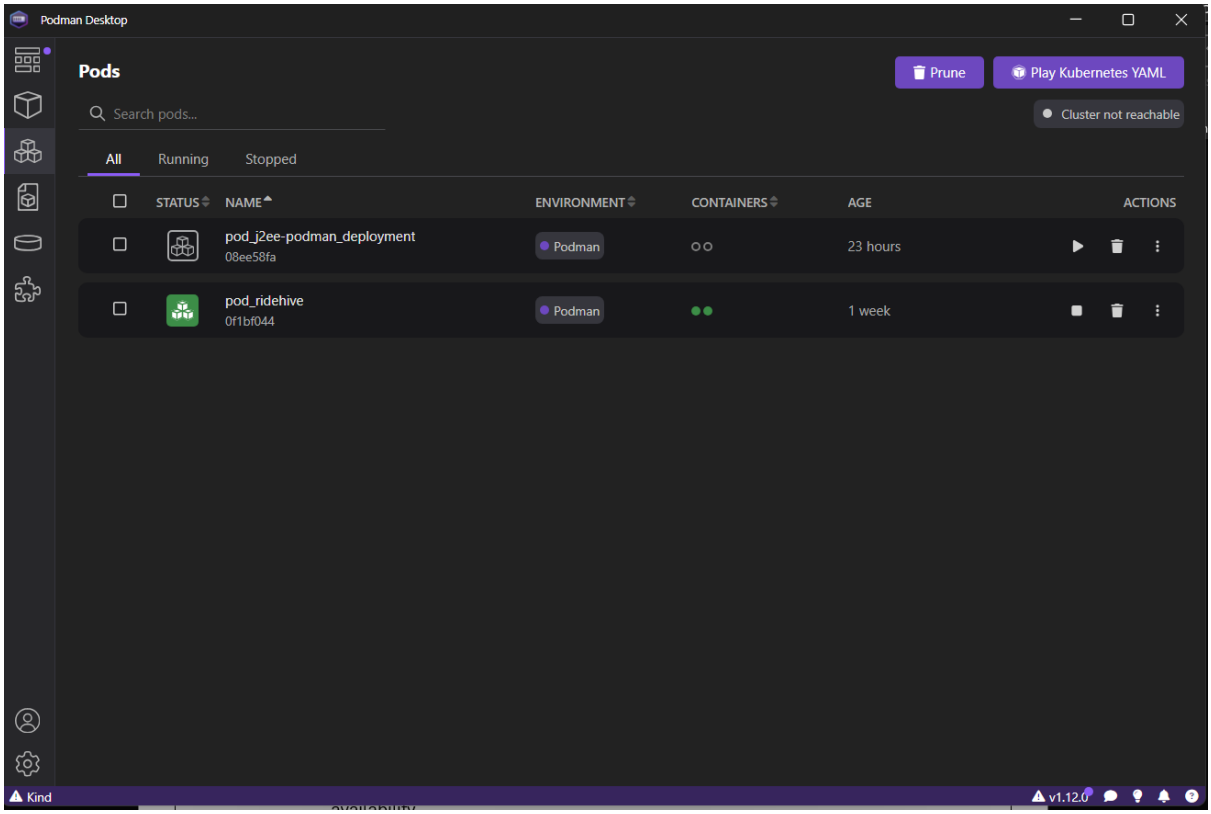
PHP-Apache: Handles the PHP application running on an Apache web server, with the project files and environment variables mounted as volumes.

MySQL Database: Configures a MySQL 8.0 database, including initialization with SQL scripts and database persistence using volumes.

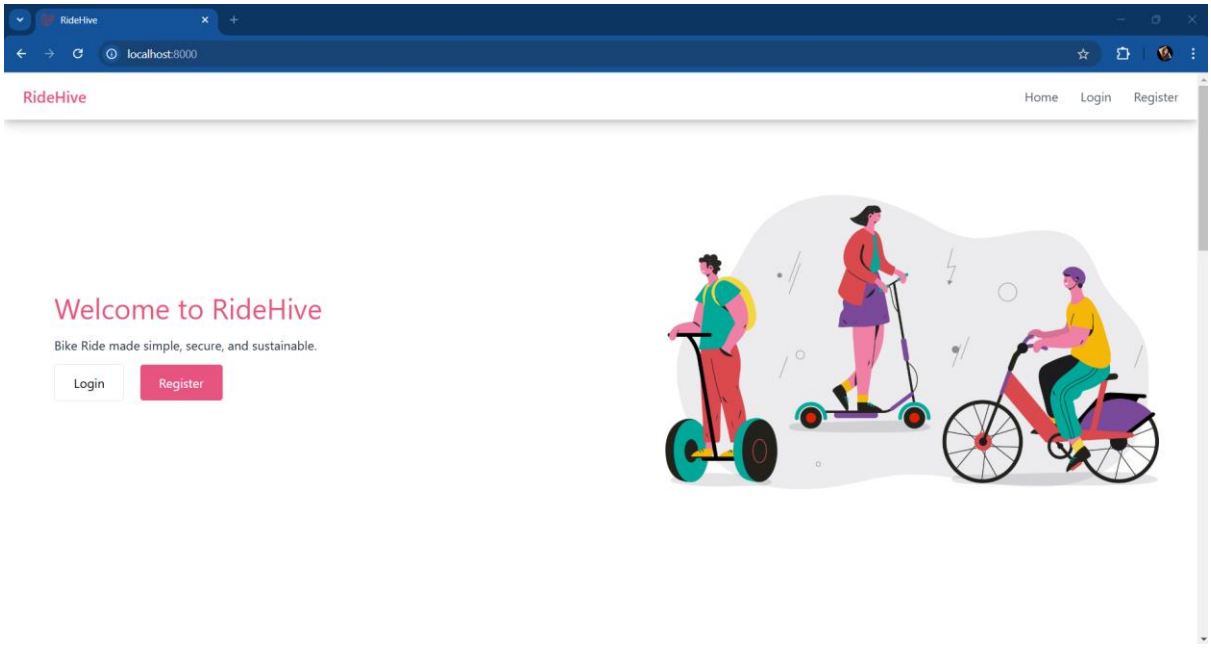
Dockerfile: Builds the PHP environment using version 8.1, installing necessary extensions such as pdo and pdo_mysql for database interaction. The source files are copied into the web server directory, and port 80 is exposed for web access.

9. Screenshots

Podman Deployment:



Home Page:



RideHive

localhost:8000

Our Features

Easy Rental

Rent a bike or scooter effortlessly.

Save Money

Reduce your transportation costs.

Eco-Friendly

Help reduce your carbon footprint by riding bikes.

RideHive

localhost:8000

What Our Users Say

"RideHive made my daily commute so much easier! It's quick, affordable, and eco-friendly."

Jhon J.

User for 3+ years

"I love that I can find rental bikes nearby. It's both cost-effective and convenient!"

Sarah D.

User for 1+ year

"Being part of RideHive has helped me reduce my travel expenses while helping the environment!"

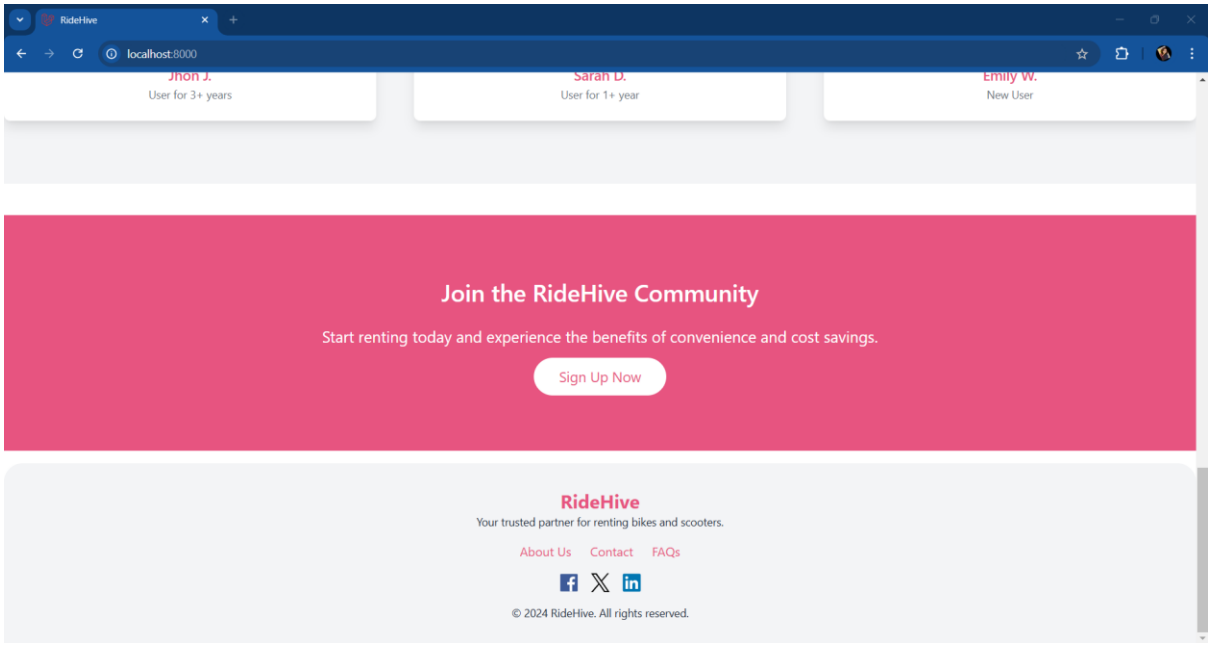
Emily W.

New User

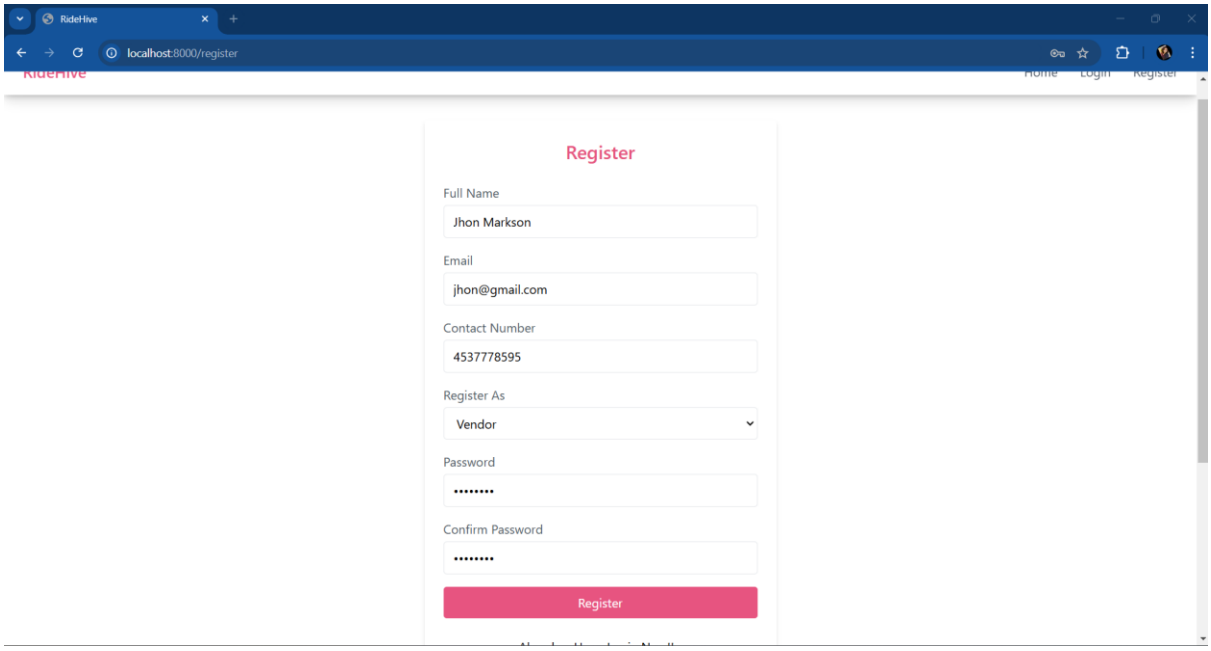
Join the RideHive Community

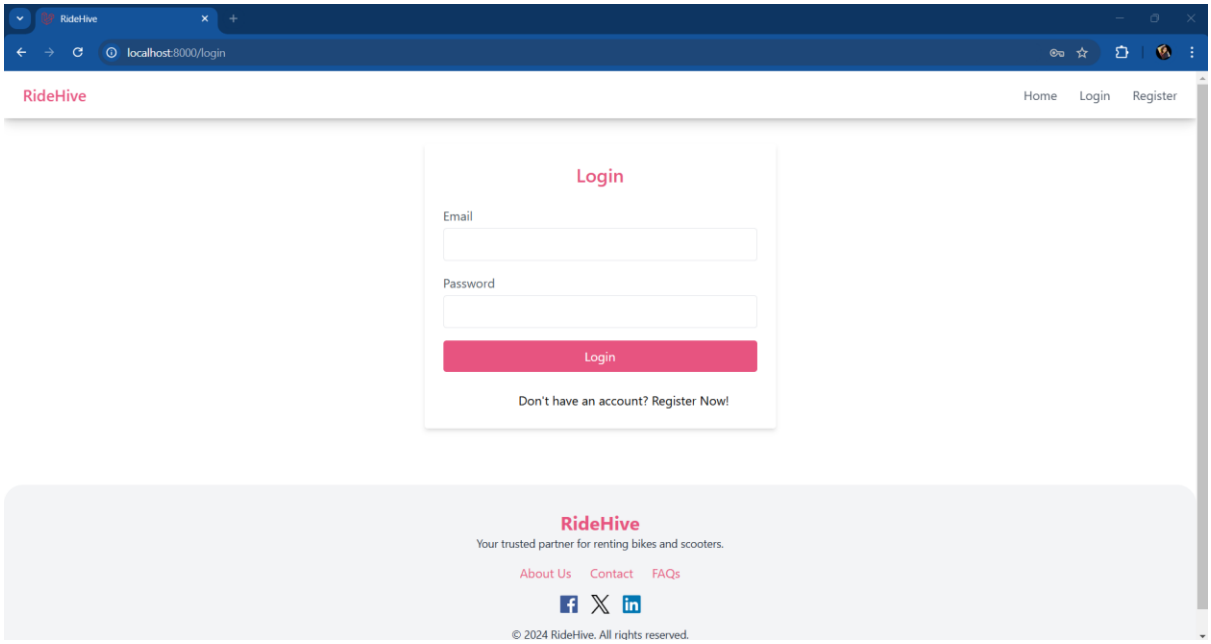
Start renting today and experience the benefits of convenience and cost savings.

Sign Up Now

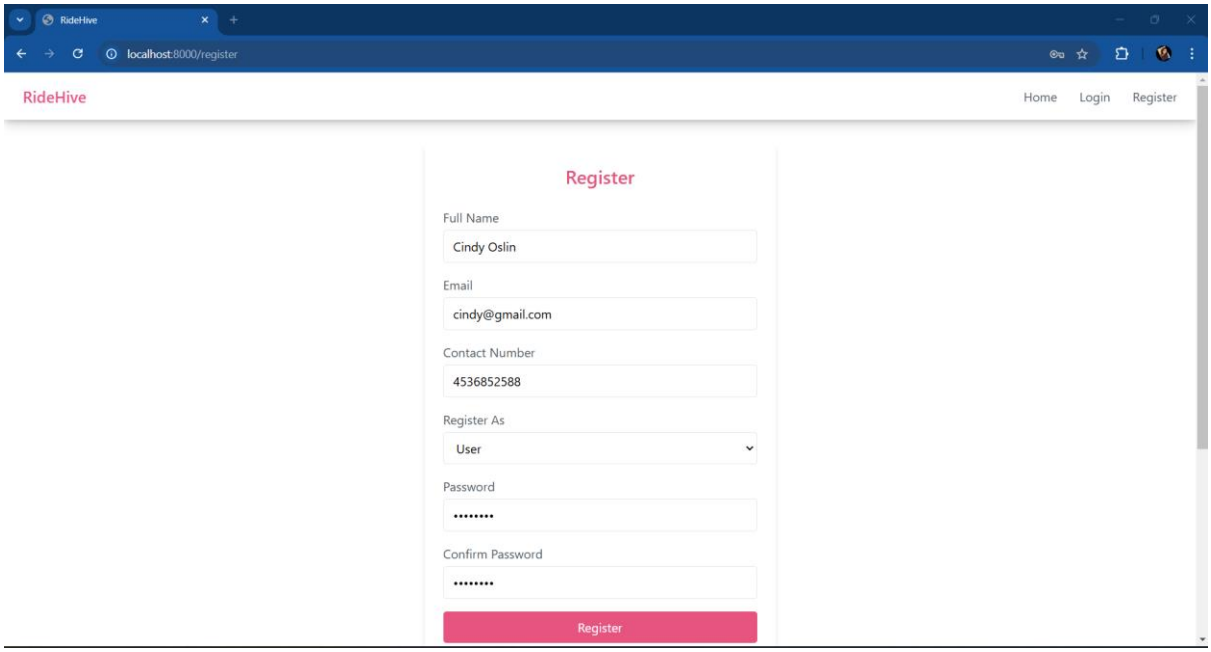


Register as Vendor:

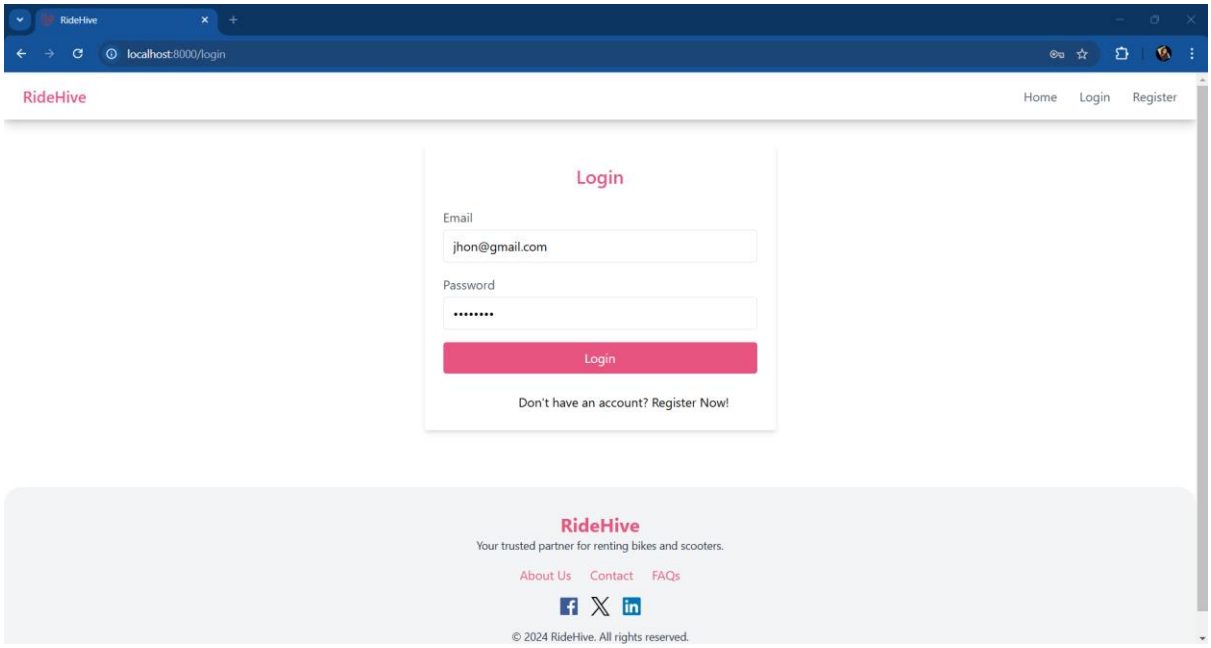




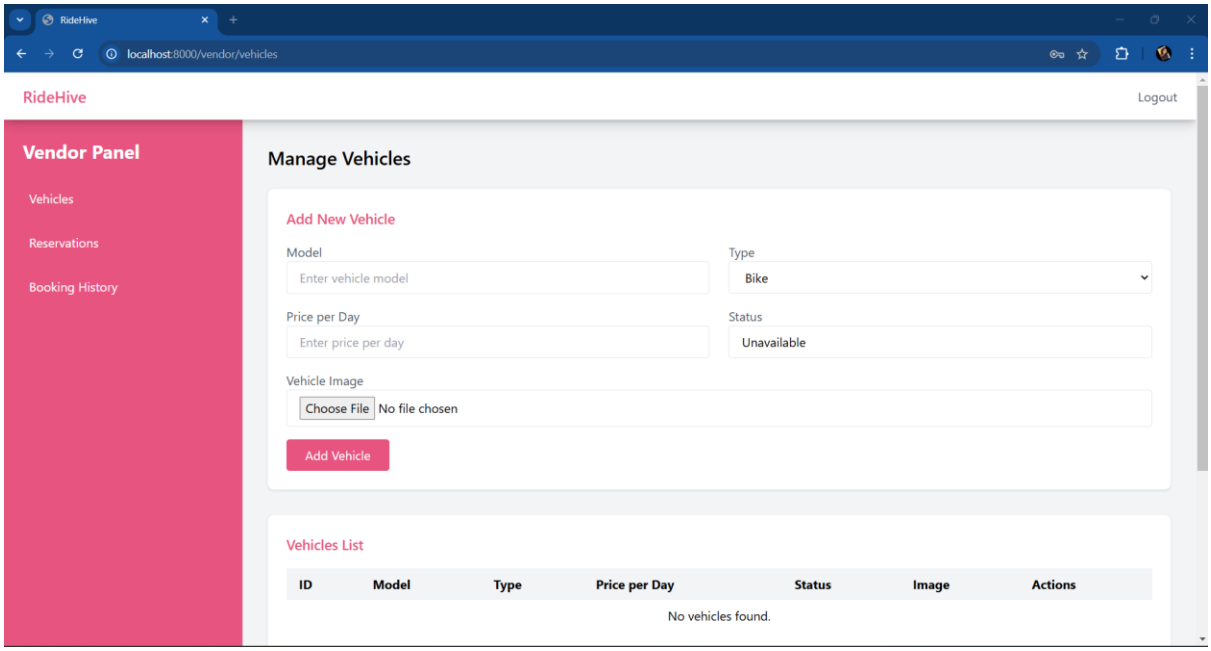
Register as User:



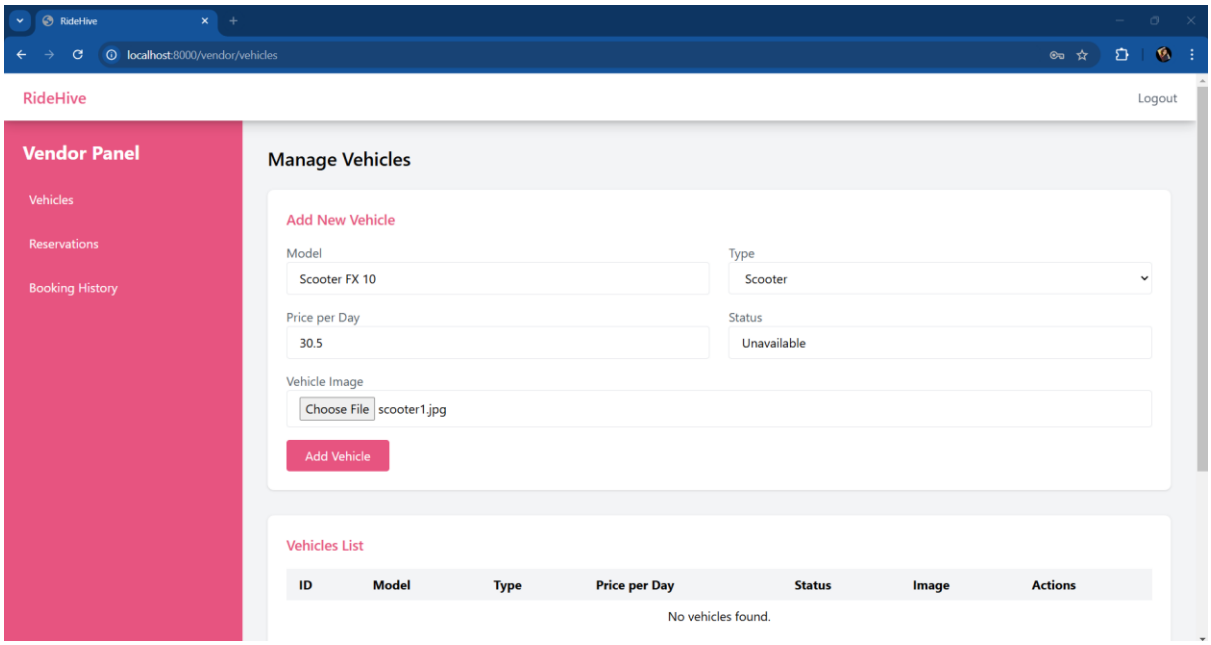
Login:

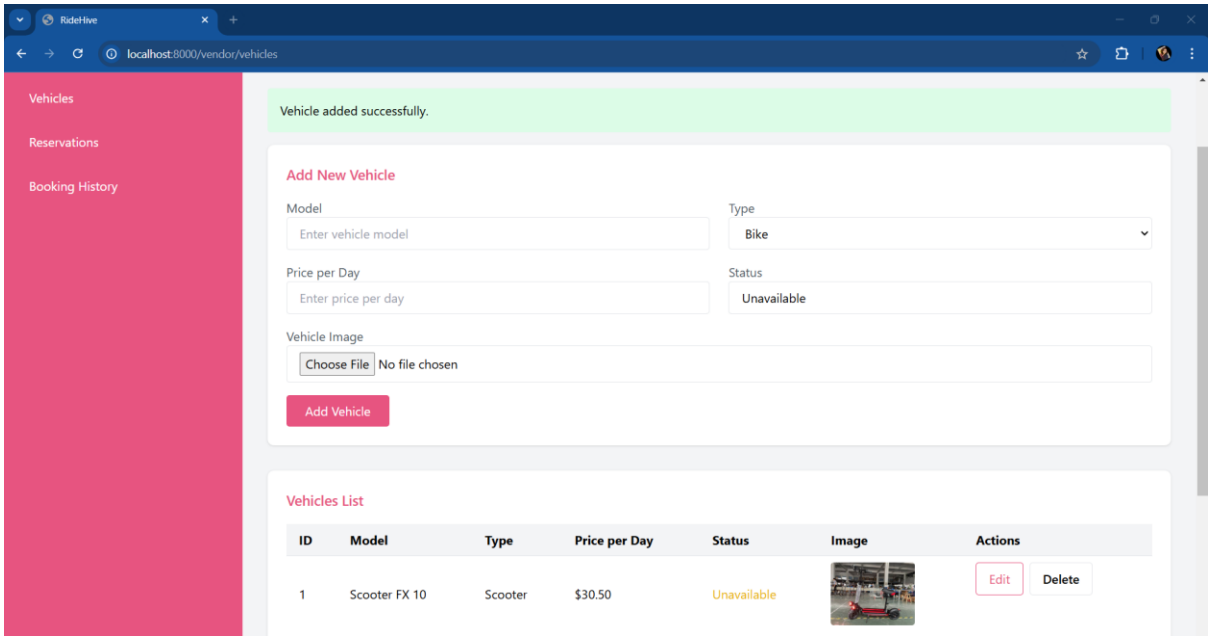


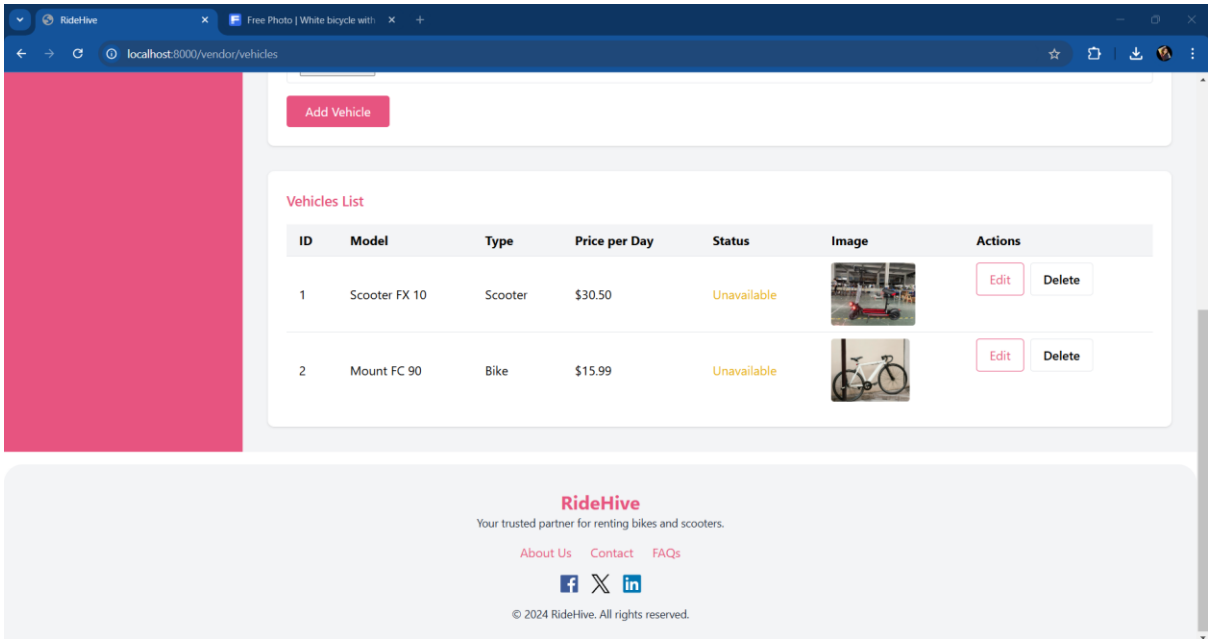
Vendor Panel:



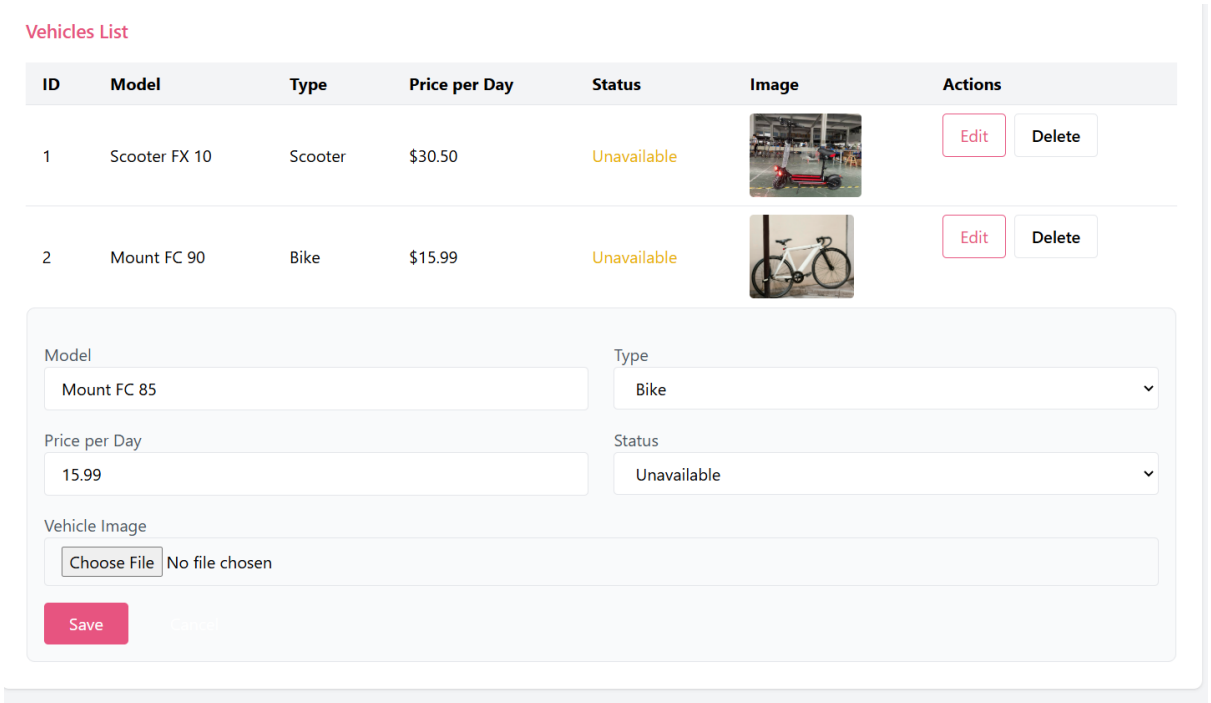
Adding Vehicle:

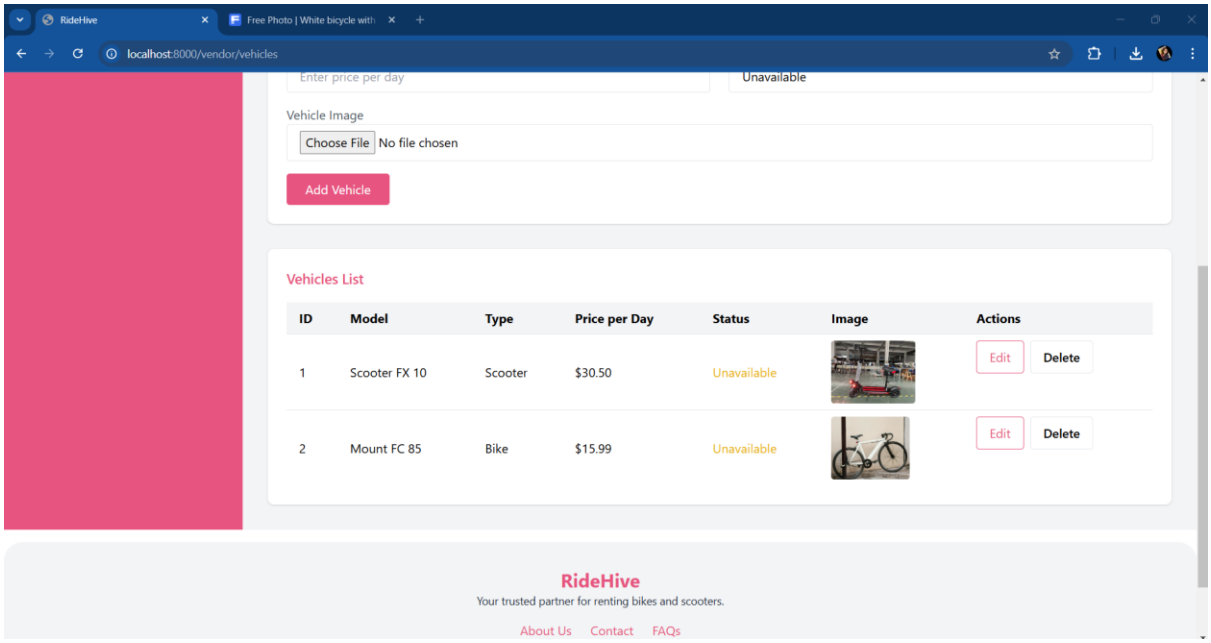




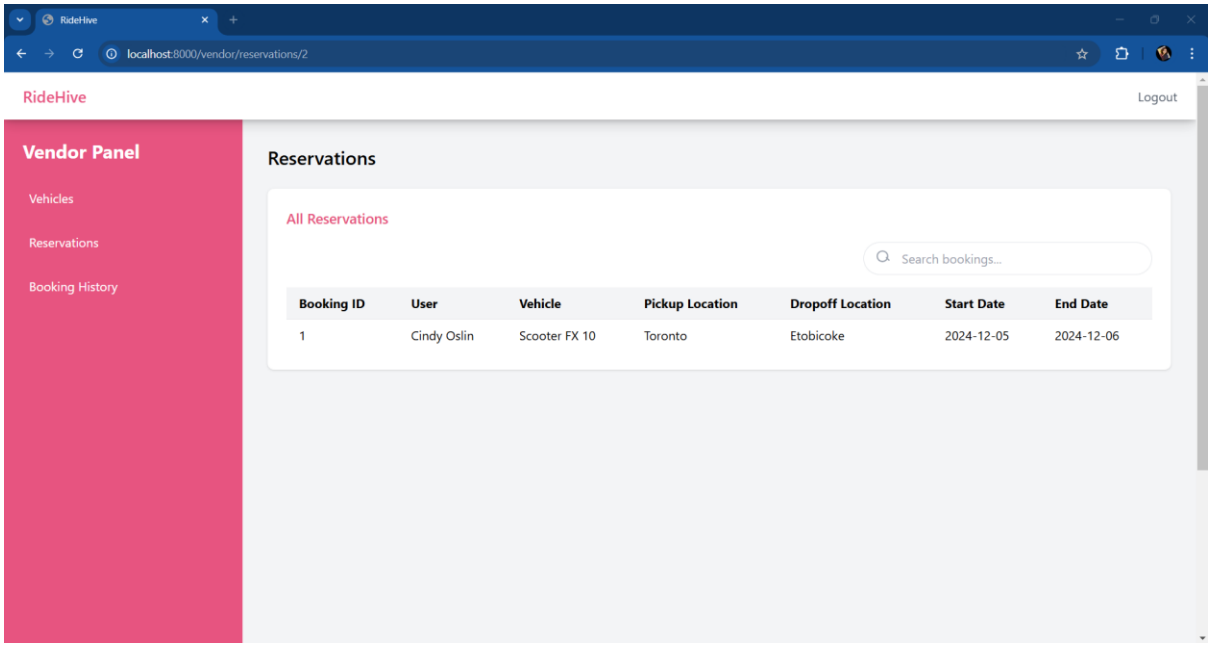


Edit Vehicle:

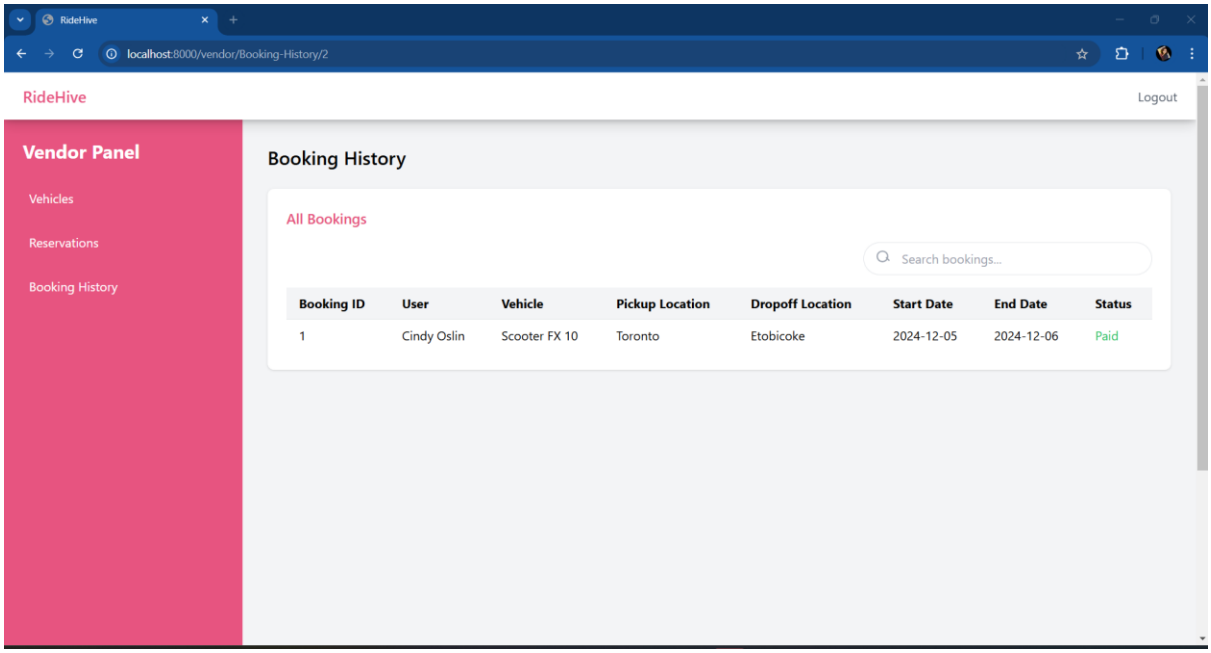




Reservations:

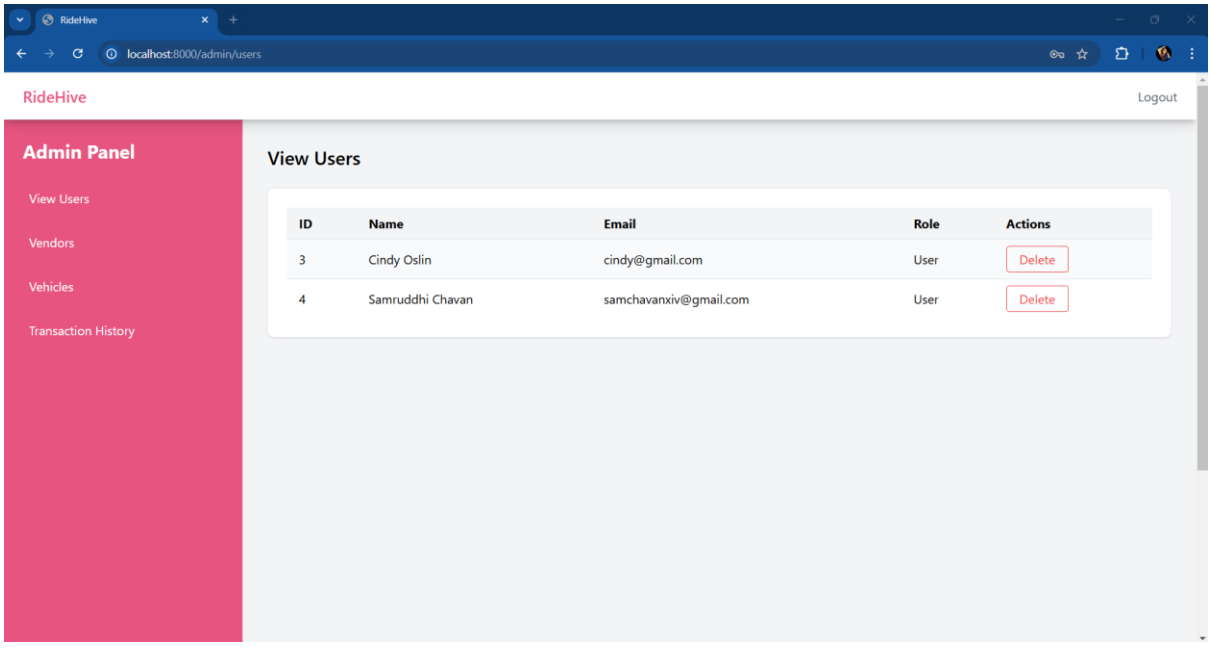


Booking History:

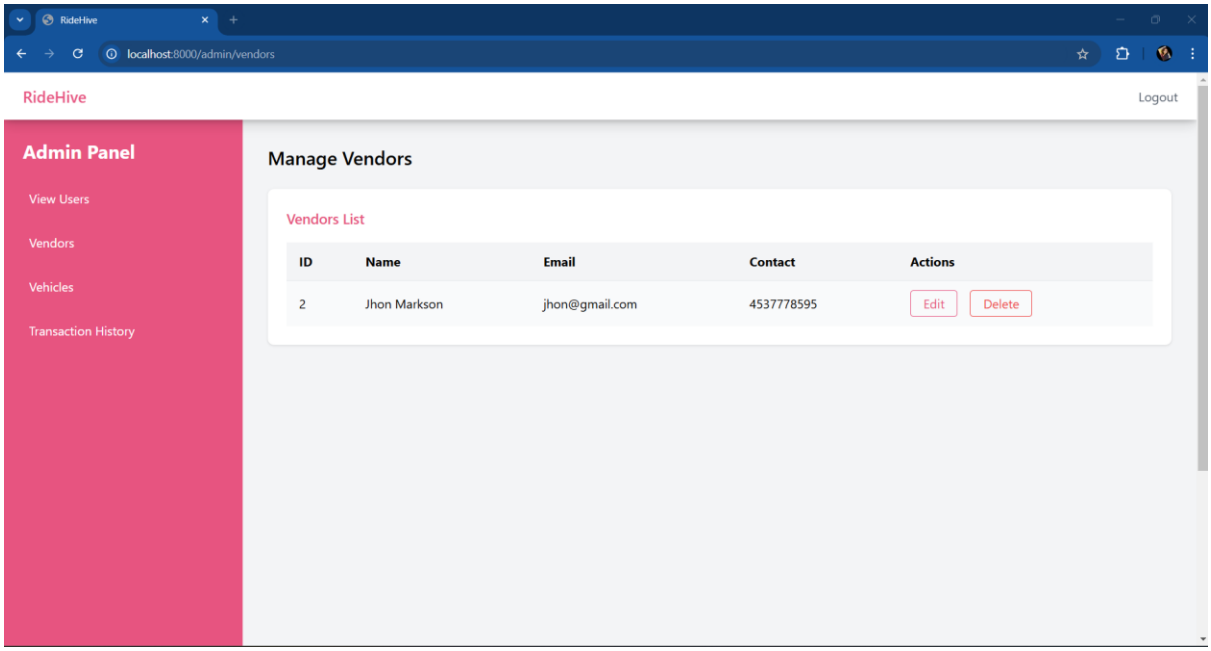


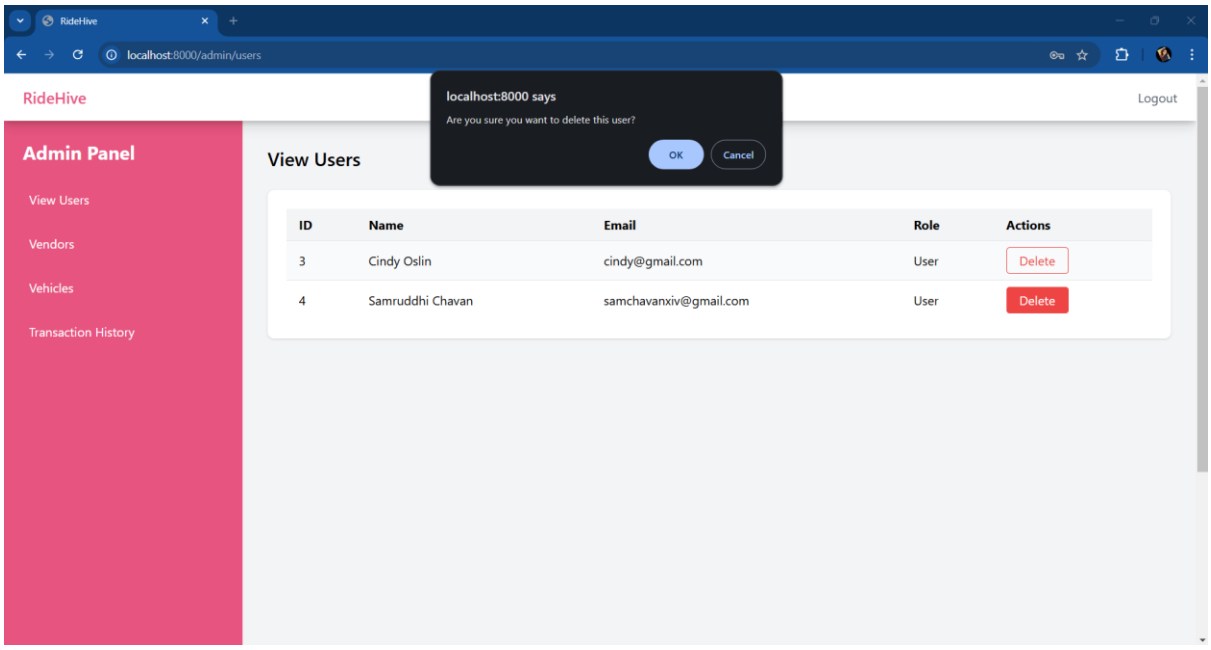
Admin Panel:

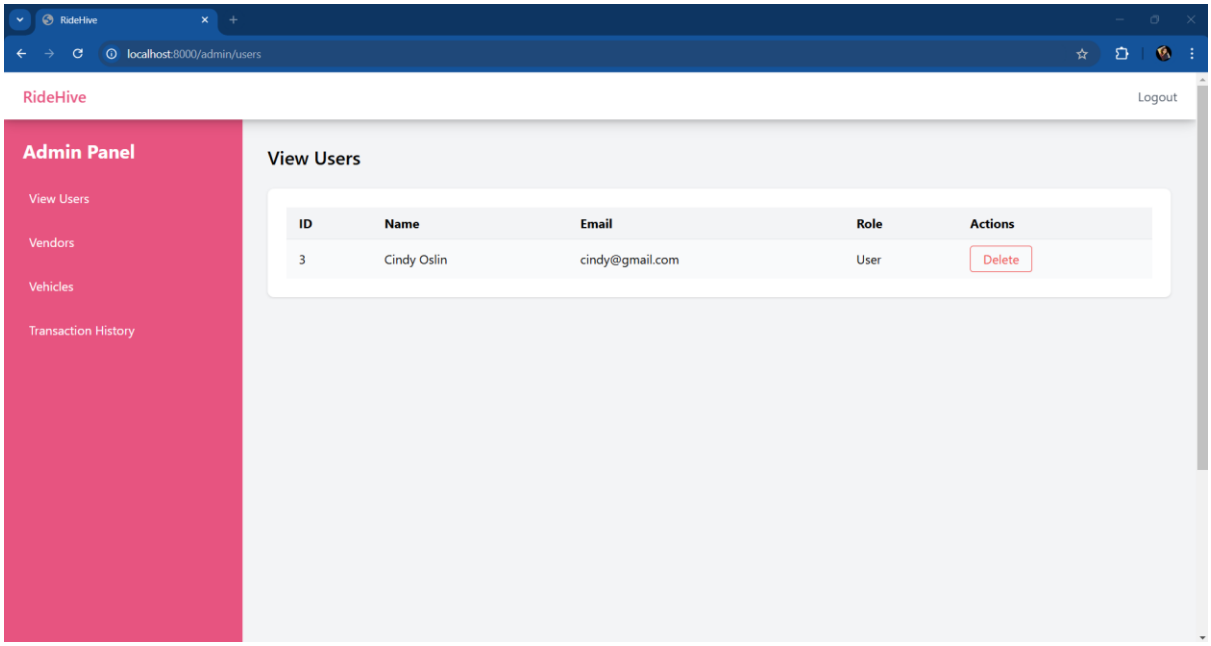
User Management



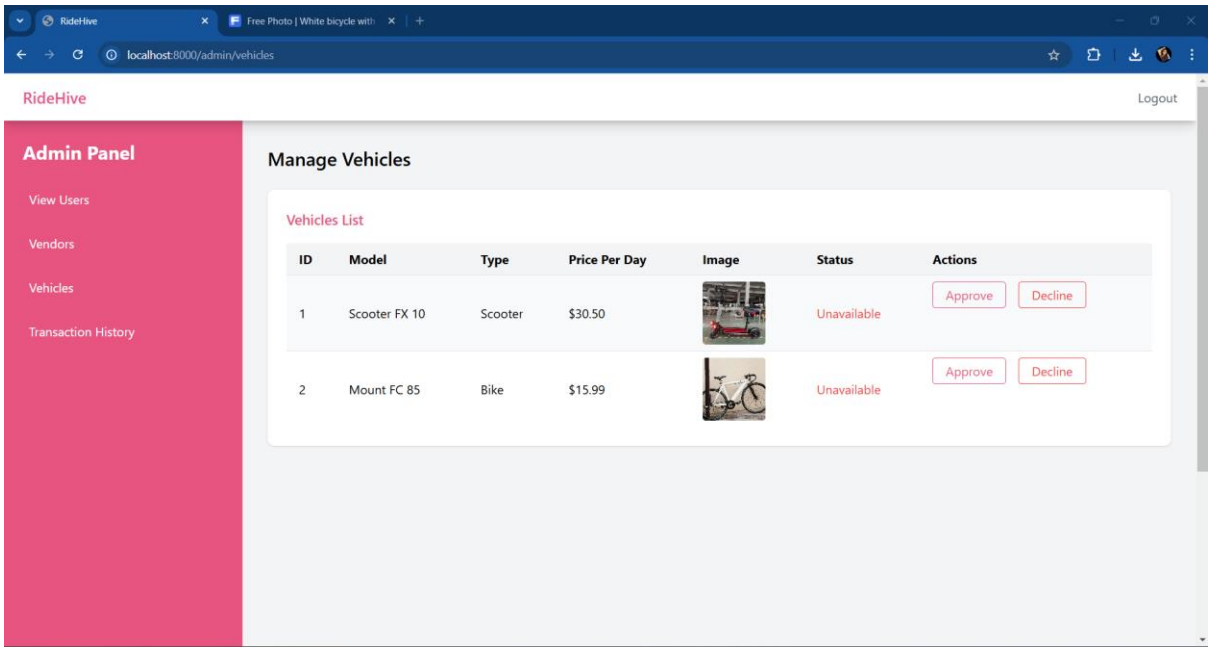
Vendor Management



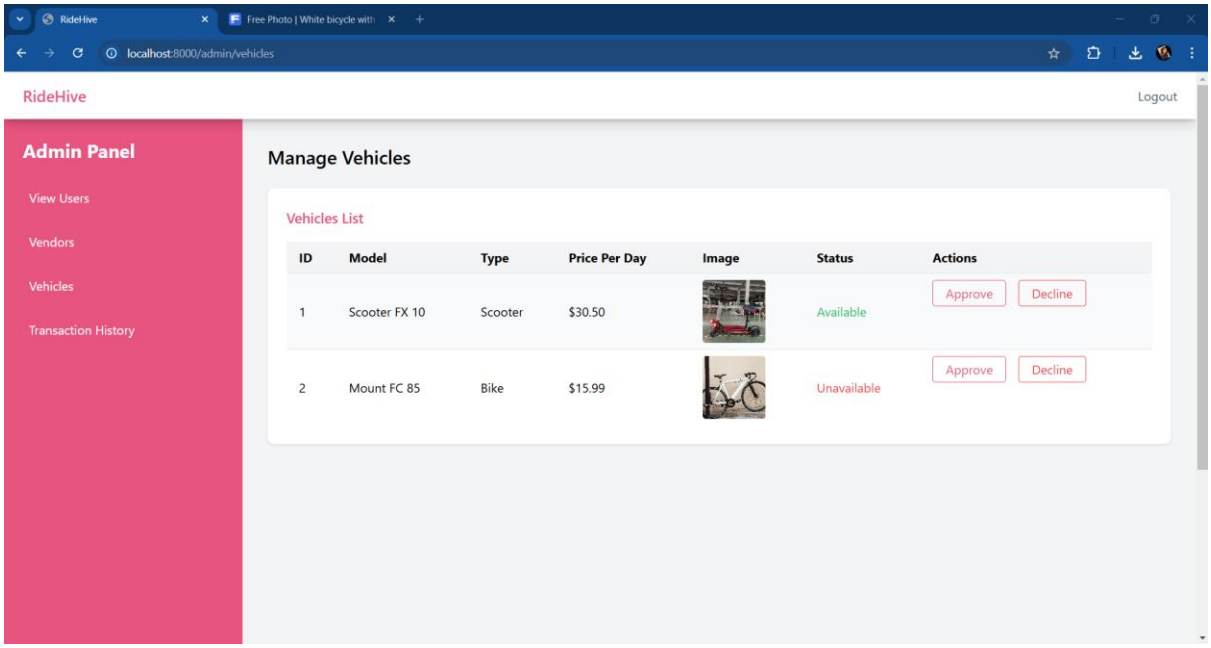




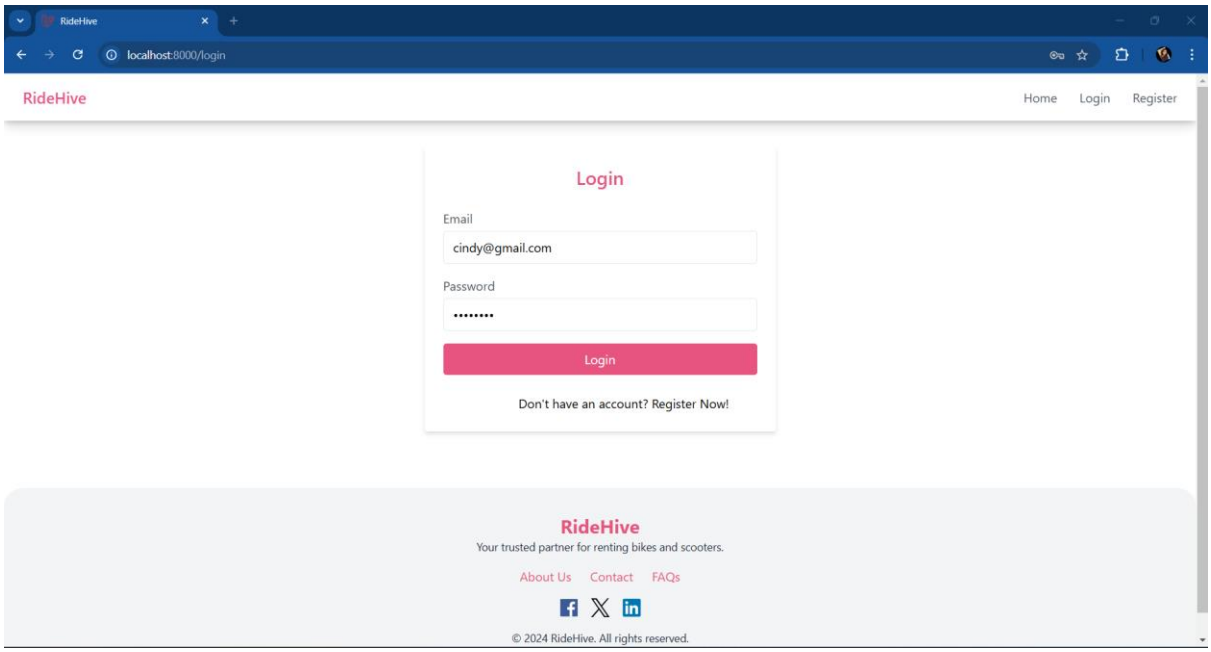
Admin Vehicle Approval Process:



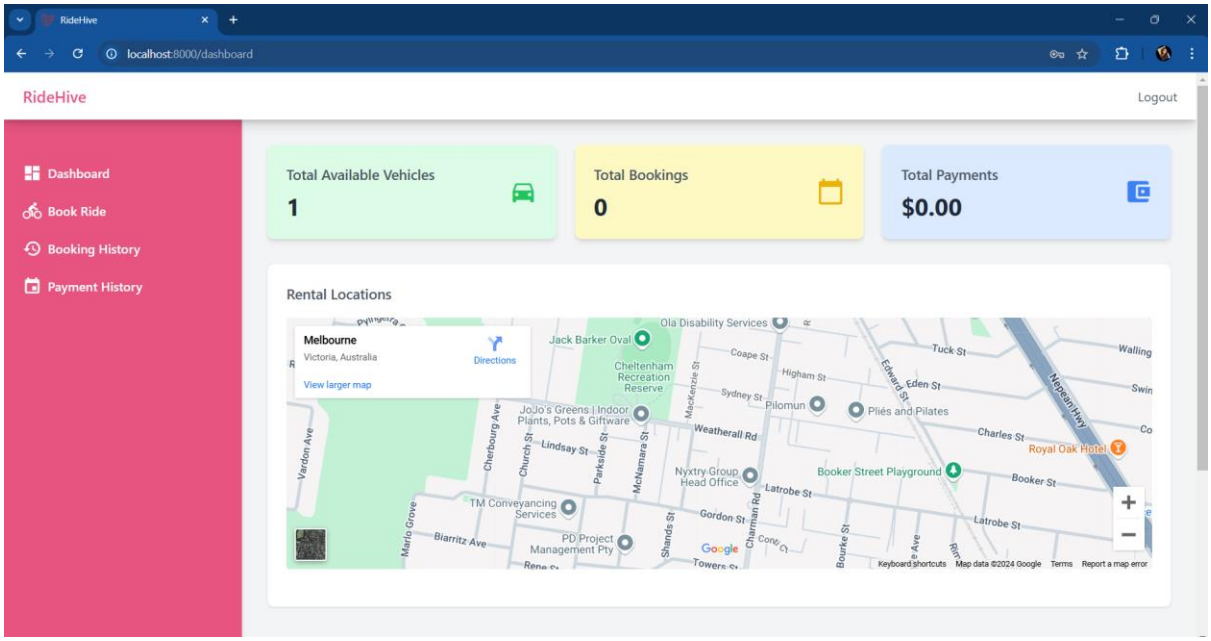
Transaction History:



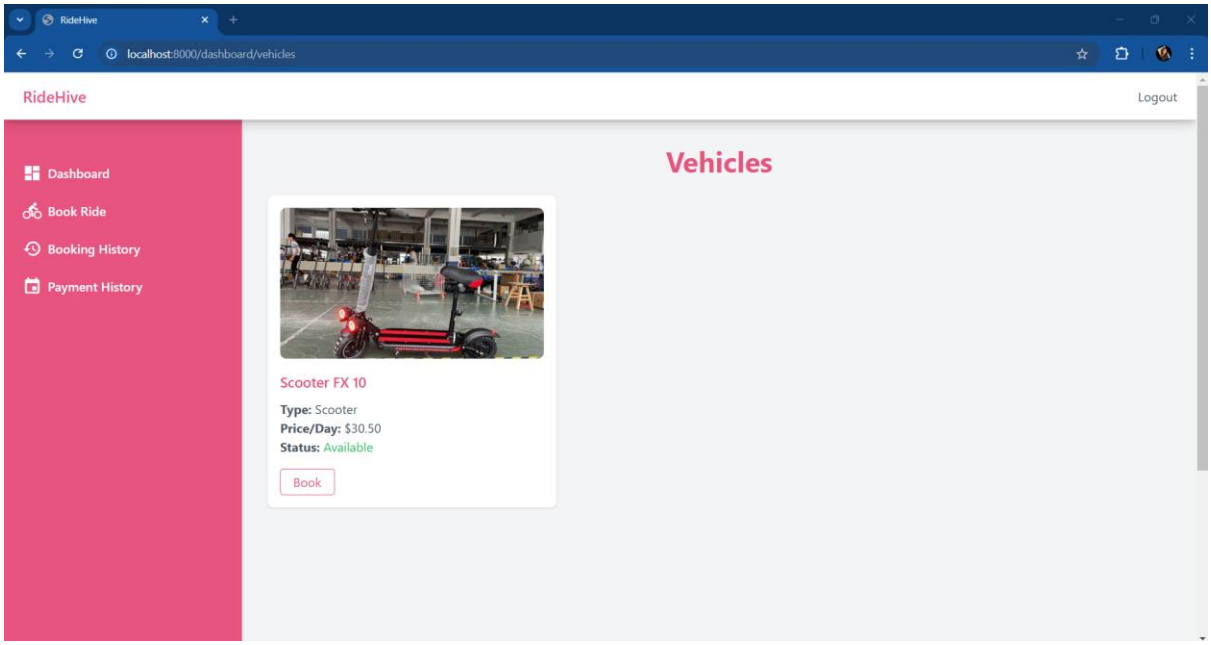
Login as User:



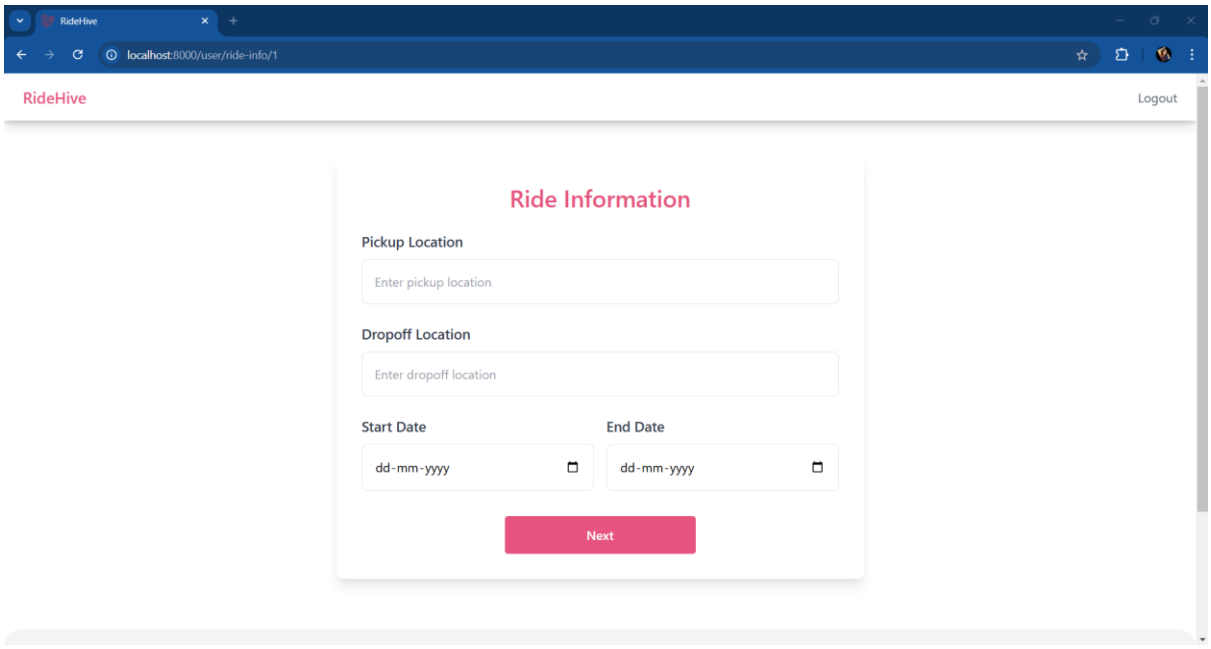
User Dashboard

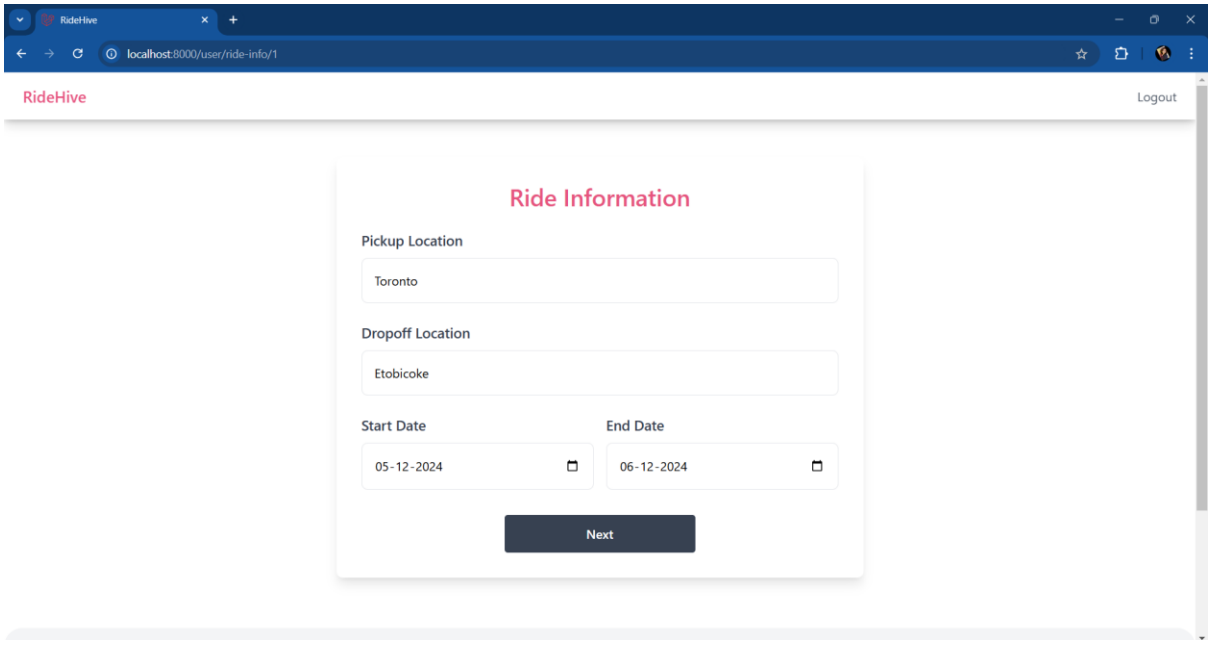


Available Vehicle Visible to user:

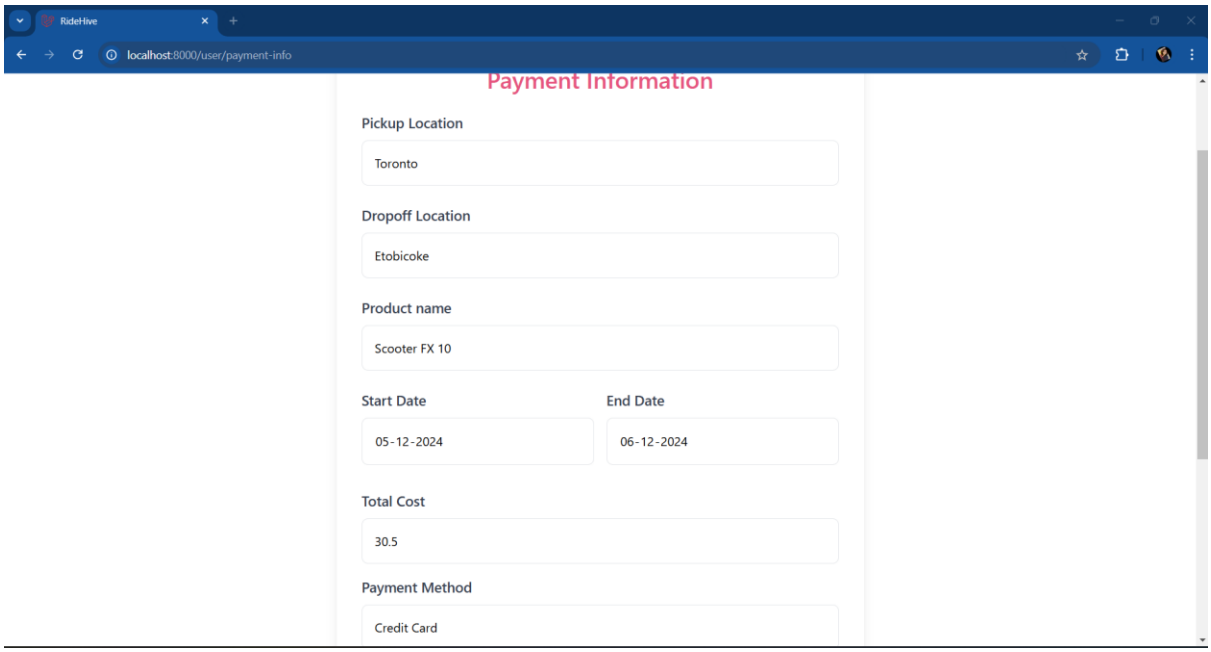


Booking a Scooter:

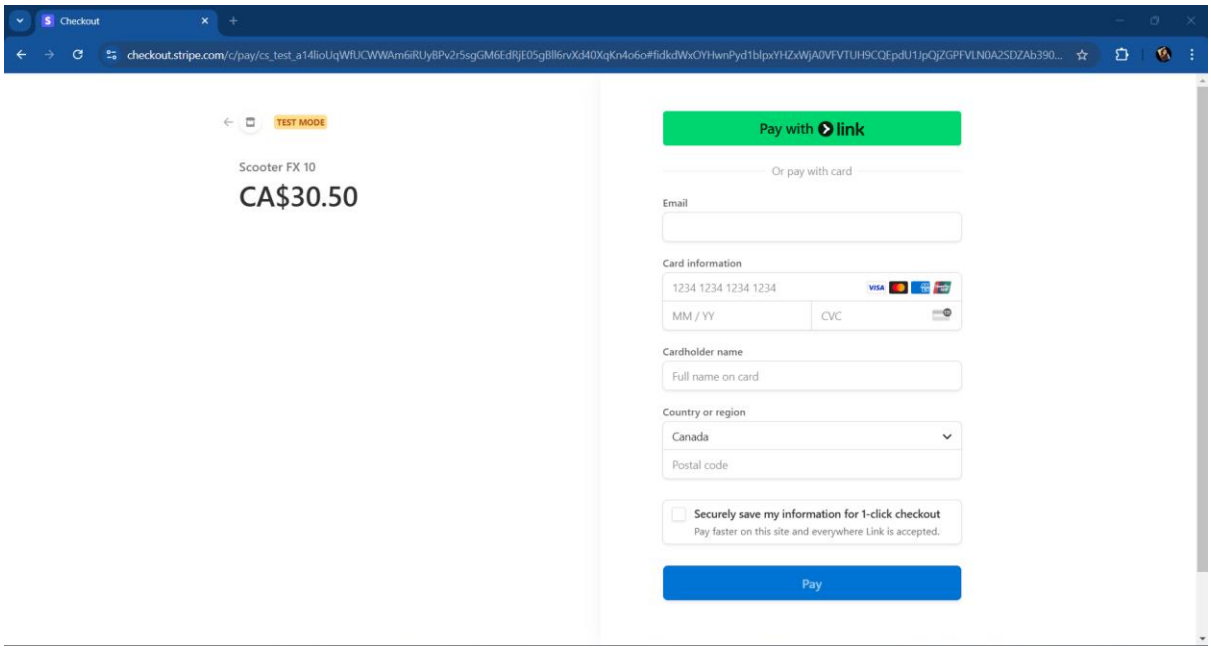


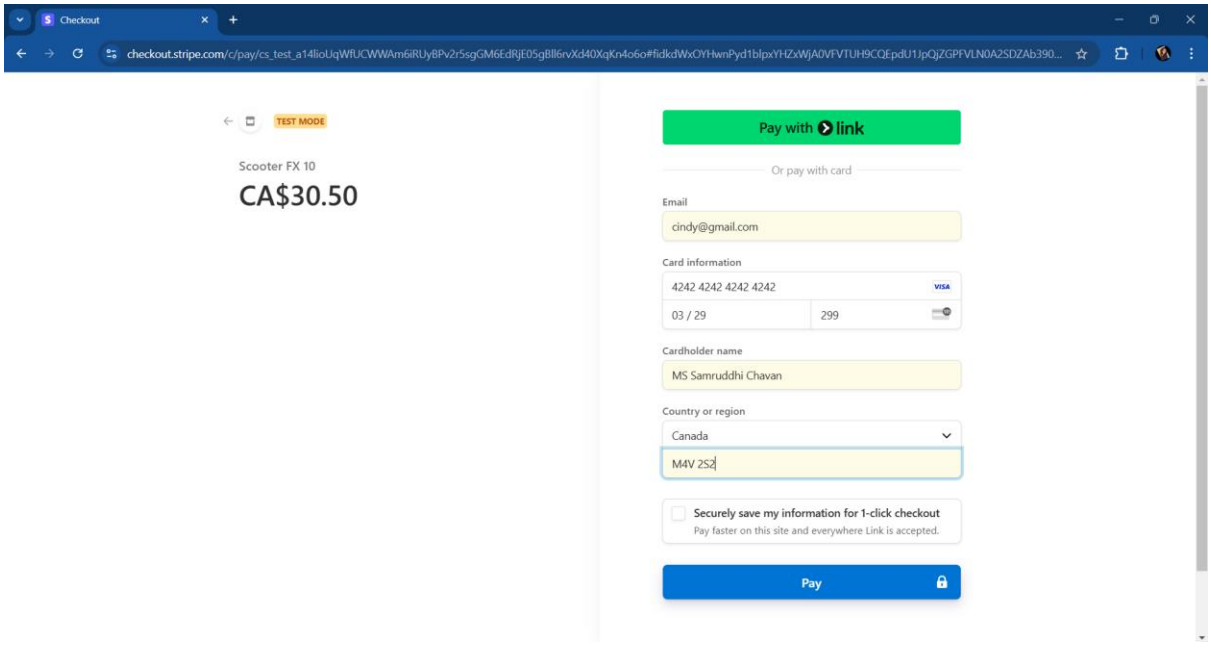


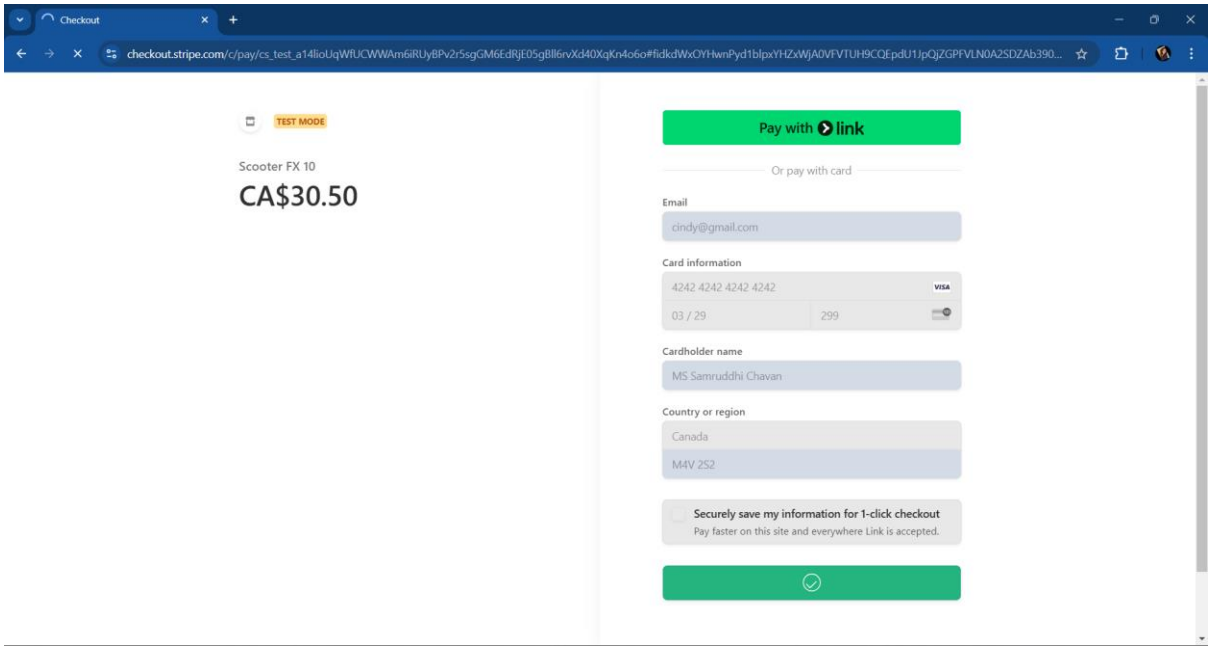
Payment Information:

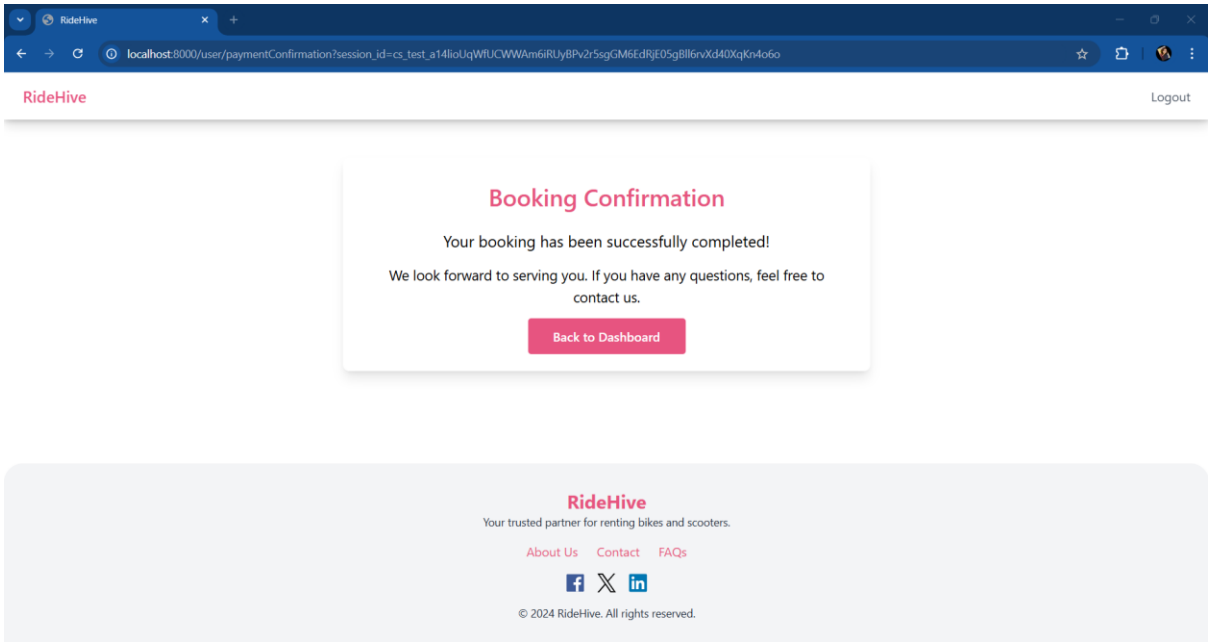


3rd – Party API Integration with STRIPE:

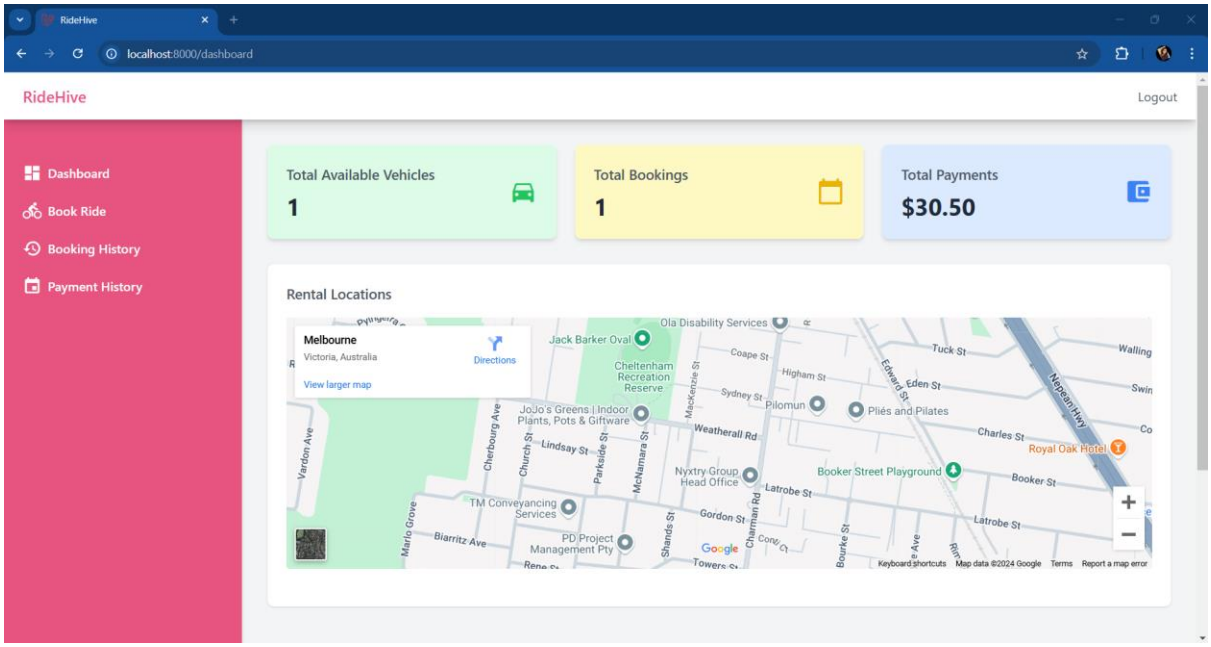



Checkout
checkout.stripe.com/c/pay/cs_test_a14lioUqWfUCWWAm6iRUyBPv2r5sgGM6EdRjE05gBll6rvXd40XqKn4o6o#fidkdWxOYHwnPyd1blpxYHZxWjA0VFVTUH9CQEpdU1JpQjZGPFVLN0A2SDZAb390...
TEST MODE
Scooter FX 10
CA$30.50
Pay with link
Or pay with card
Email
cindy@gmail.com
Card information
4242 4242 4242 4242
03 / 29
299
Cardholder name
MS Samruddhi Chavan
Country or region
Canada
M4V 2S2
Securely save my information for 1-click checkout
Pay faster on this site and everywhere Link is accepted.
Pay

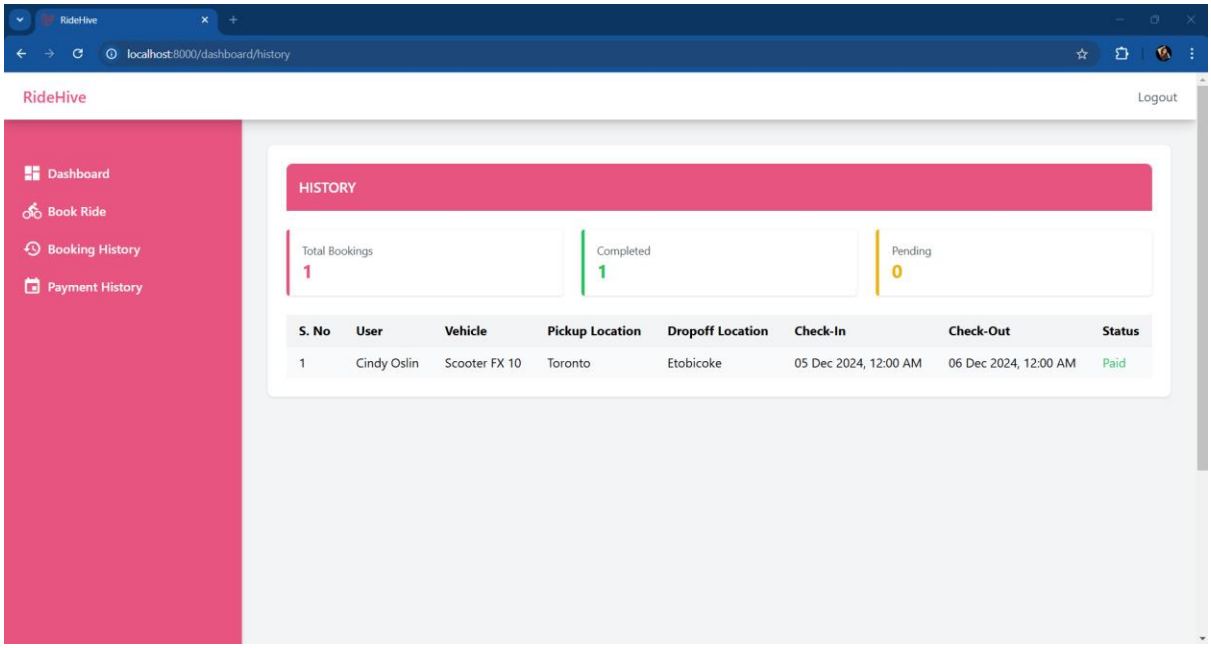

Checkout
checkout.stripe.com/c/pay/cs_test_a14lioUqWfUCWWAm6iRUyBPv2r5sgGM6EdRjE05gBll6rvXd40XqKn4o6o#fidkdWxOYHwnPyd1blpxYHZxWjA0VFVTUH9CQEpdU1JpQjZGPFVLN0A2SDZAb390...
TEST MODE
Scooter FX 10
CA$30.50
Pay with link
Or pay with card
Email
cindy@gmail.com
Card information
4242 4242 4242 4242
03 / 29
299
Cardholder name
MS Samruddhi Chavan
Country or region
Canada
M4V 2S2
Securely save my information for 1-click checkout
Pay faster on this site and everywhere Link is accepted.



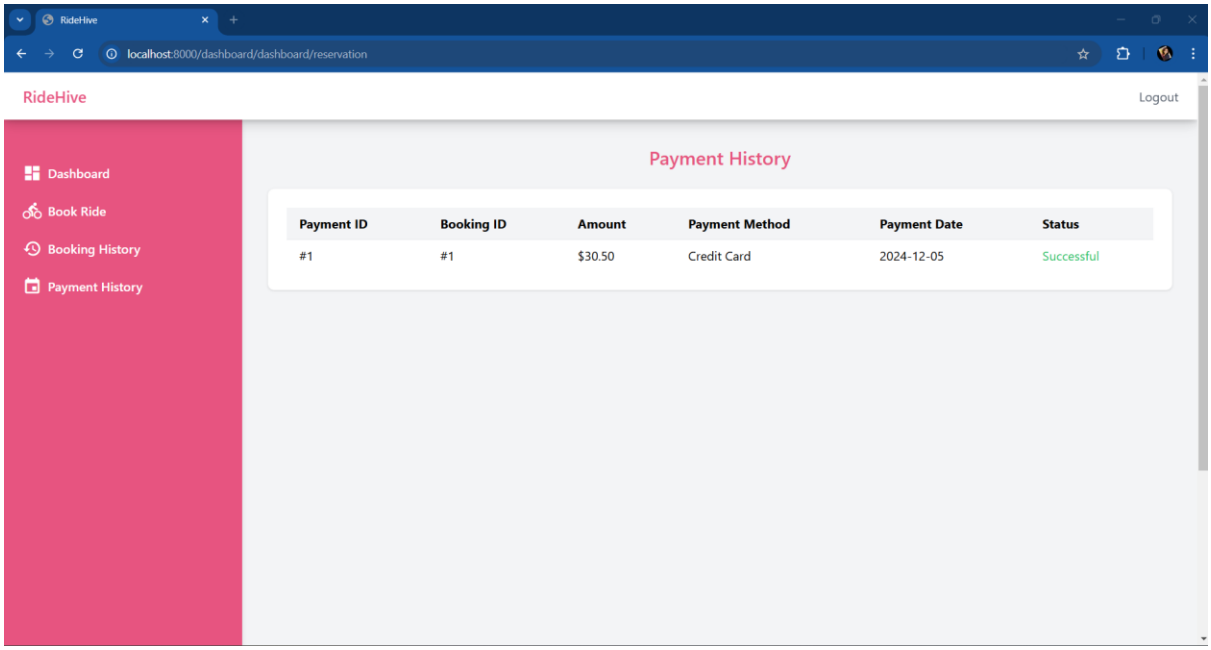
Updated Dashboard



Booking History:

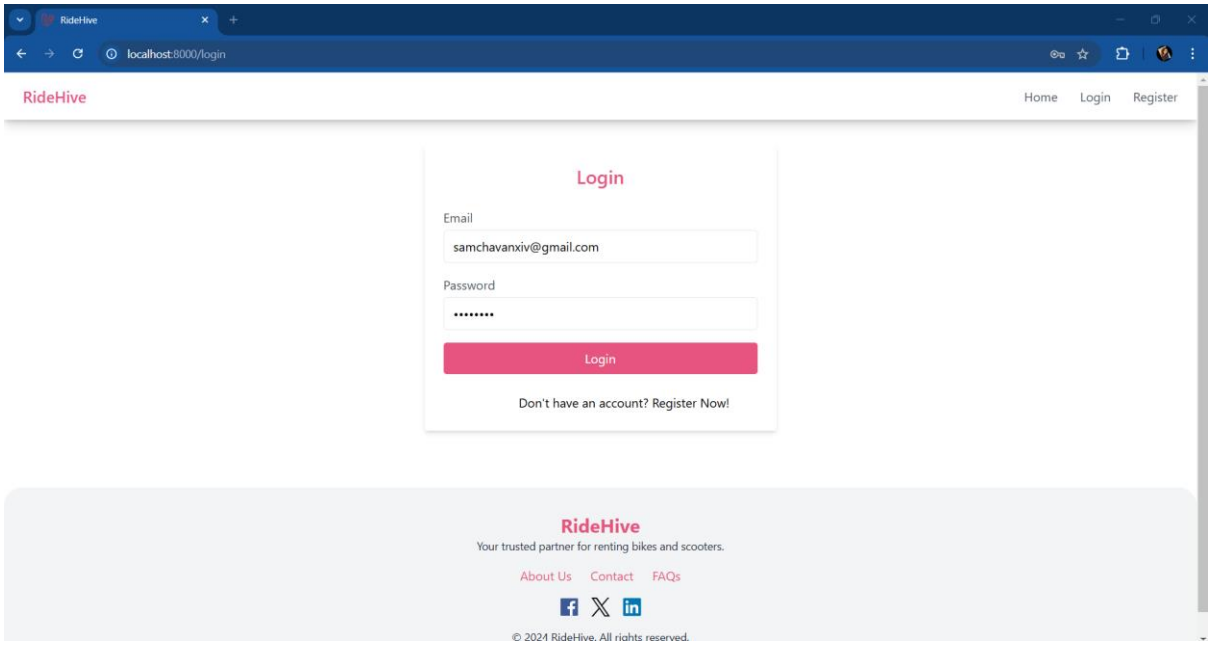


Payment History

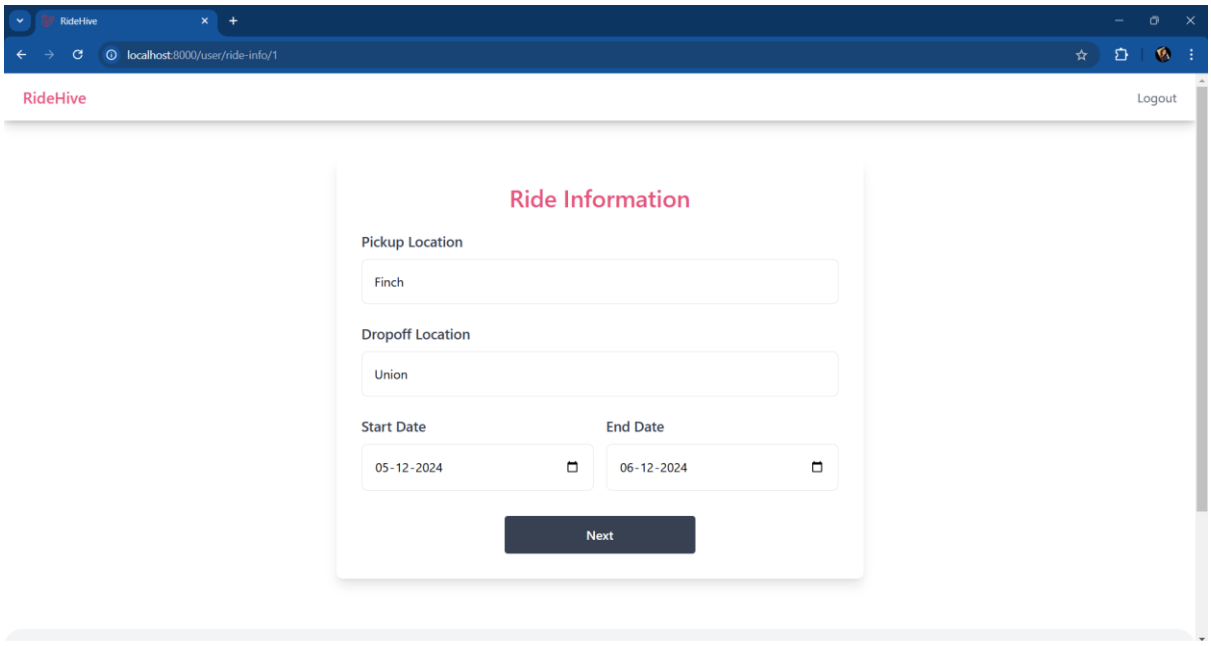


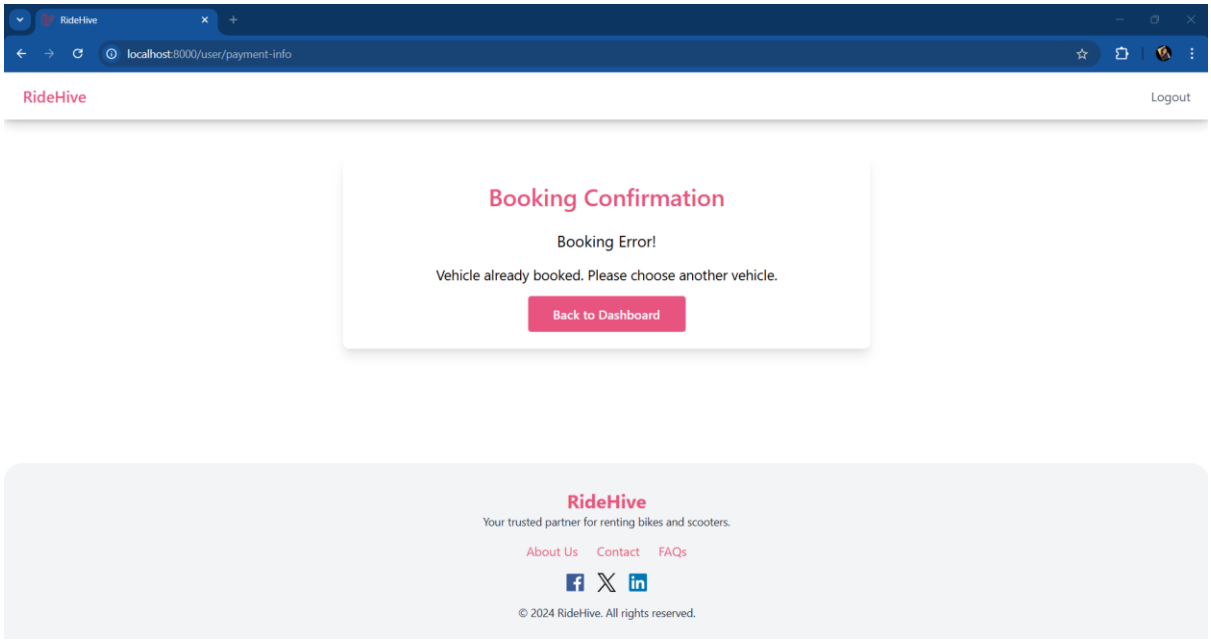
Handling Overlap: Booking a Vehicle Already Reserved for the Requested Period

Login as New User:



Book the Scooter:




RideHive
localhost:8000/user/payment-info
RideHive
Logout
Booking Confirmation
Booking Error!
Vehicle already booked. Please choose another vehicle.
Back to Dashboard
RideHive
Your trusted partner for renting bikes and scooters.
About Us
Contact
FAQs
© 2024 RideHive. All rights reserved.

10. Conclusion

The RideHive Bike and Scooter Rental System is a well-designed platform addressing the growing need for affordable and flexible urban transportation. It provides a seamless experience for users to register, log in, search, and book vehicles, while vendors can list and manage their fleets efficiently. The integration of Stripe ensures secure online payment processing, while Laravel's MVC architecture enhances modularity and scalability. The database is robustly structured, linking users, vehicles, bookings, payments to maintain operational efficiency. Additionally, an admin dashboard facilitates comprehensive monitoring and management of users, vendors, transactions, and bookings. By leveraging modern technologies like Laravel, MySQL, and Tailwind CSS, RideHive is equipped to handle current demands and future enhancements, making it a scalable and secure rental solution.